Carrier Ethernet Configuration Guide, Cisco IOS XE Everest 16.6

First Published: 2017-07-31

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 8

CHAPTER 9

CHAPTER 19

CHAPTER 20

CHAPTER 1

Read Me First

Important Information about Cisco IOS XE 16

Effective Cisco IOS XE Release 3.7.0E for Catalyst Switching and Cisco IOS XE Release 3.17S (for Access and Edge Routing) the two releases evolve (merge) into a single version of converged release—the Cisco IOS XE 16—providing one release covering the extensive range of access and edge products in the Switching and Routing portfolio.

Feature Information

Use Cisco Feature Navigator to find information about feature support, platform support, and Cisco software image support. An account on Cisco.com is not required.

Related References

- Cisco IOS Command References, All Releases

Obtaining Documentation and Submitting a Service Request

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

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.

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.

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

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.

  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.

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:

- 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 .

High Availability Features Supported by 802.3ah

In access and service provider networks using Ethernet technology, High Availability (HA) is a requirement, especially on Ethernet 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 Switch Processor (RSP) (a standby RSP that has the same software image as the active RSP and supports synchronization of line card, protocol, and application state information between RSPs for supported features and protocols). End-to-end connectivity status is maintained on the CE, PE, and access aggregation PE (uPE) network nodes based on information received by protocols such as 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 (for example, CFM and 802.3ah) maintain configuration data and dynamic data, which is learned through protocols. Every transaction involves either accessing or updating data among the various databases. If the databases are synchronized across active and standby modules, the RSPs are transparent to clients.

Cisco infrastructure provides various component application program interfaces (APIs) for clients that are helpful in maintaining a hot standby RSP. Metro Ethernet HA clients (such as, HA/ISSU, CFM HA/ISSU, 802.3ah HA/ISSU) interact with these components, update the databases, and trigger necessary events to other components.

Benefits of 802.3ah 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 delivering higher service levels due to the elimination of network downtime during upgrades

NSF SSO Support in 802.3ah OAM

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

For detailed information about the SSO feature, see the “Configuring Stateful Switchover” module of the *High Availability Configuration Guide*. For detailed information about the NSF feature, see the “Configuring Cisco Nonstop Forwarding” module of the *High Availability Configuration Guide.*

ISSU Support in 802.3ah OAM

Cisco In-Service Software Upgrades (ISSUs) allow you to perform a Cisco software upgrade or downgrade without disrupting packet flow. ISSU is automatically enabled in 802.3ah. OAM performs a bulk update and a runtime update of the continuity check database to the standby Route Switch Processor (RSP), including adding, deleting, or updating a row. This checkpoint data requires ISSU capability to transform messages from one release to another. All the components that perform active RSP to standby RSP updates using messages require ISSU support.

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 “Performing an In Service Software Upgrade” module of the *High Availability Configuration Guide*.

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]
5. **no ethernet oam link-monitor supported**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam link-monitor supported**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]
5. **no ethernet oam link-monitor on**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam link-monitor on**
5. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]
5. **ethernet oam link-monitor frame** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}
6. **ethernet oam link-monitor frame-period** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *frames*}
7. **ethernet oam link-monitor frame-seconds** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}
8. **ethernet oam link-monitor receive-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}
9. **ethernet oam link-monitor transmit-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}
10. **ethernet oam link-monitor symbol-period** {**threshold** {**high** {**none** | *high-symbols*} | **low** *low-symbols*} | **window** *symbols*}
11. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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 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 6	**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 7	**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.

	Command or Action	Purpose

Step 8	**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 9	**ethernet oam link-monitor transmit-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}
	Example:
	`Device(config-if)# 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 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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template** *template-name*
4. **ethernet oam link-monitor receive-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}
5. **ethernet oam link-monitor transmit-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}

6. **ethernet oam link-monitor symbol-period** {**threshold** {**high** {**none** | *high-symbols*} | **low** *low-symbols*} | **window** *symbols*}
7. **ethernet oam link-monitor frame** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}
8. **ethernet oam link-monitor frame-period** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *frames*}
9. **ethernet oam link-monitor frame-seconds** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}
10. **exit**
11. **interface** *type number*
12. **source template** *template-name*
13. **exit**
14. **exit**
15. **show running-config**

DETAILED STEPS

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

	Command or Action	Purpose

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 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 8	**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 9	**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 10	**exit** **Example:** `Device(config-template)# exit`	Returns to global configuration mode.
Step 11	**interface** *type number* **Example:**	Identifies the interface on which to use the template and enters interface configuration mode.
Step 12	**source template** *template-name* **Example:** `Device(config-if)# source template oam-temp`	Applies to the interface the options configured in the template.
Step 13	**exit** **Example:**	Returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-if)# exit`	
Step 14	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 15	**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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam remote-failure** {**critical-event** | **dying-gasp** | **link-fault**} **action** { }
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Enters interface configuration mode.
Step 4	**ethernet oam remote-failure** {**critical-event** | **dying-gasp** | **link-fault**} **action** { } **Example:**	Sets the interface to the blocking state when a critical event occurs.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.

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)# exit
!
! Enable Ethernet OAM on the CE interface
!
Device(config)#
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)#
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 , 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#

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:

   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#
```

```
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#

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
```

Additional References

Related Documents

Related Topic	Document Title
Ethernet CFM	"Configuring Ethernet Connectivity Fault Management in a Service Provider Network" module in the *Carrier Ethernet Configuration Guide*
NSF SSO Support in 802.3ah OAM	"Configuring Stateful Switchover" module in the *High Availability Configuration Guide* and "Configuring Nonstop Forwarding" in the *High Availability Configuration Guide*
ISSU Support in 802.3ah OAM	"Configuring In Service Software Upgrades" module in the *High Availability Configuration Guide*
Carrier Ethernet commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*
Configuring CFM over an EFP Interface with the Cross Connect feature on the Cisco ASR 903 Router	*Configuring the CFM over EFP Interface with Cross Connect Feature*

Related Topic	Document Title
Configuring Ethernet Virtual Connections on the Cisco ASR 903 Router	*Configuring Ethernet Virtual Connections on the Cisco ASR 903 Router*

Standards

Standard	Title
IEEE Draft P802.3ah/D3.3	*Ethernet in the First Mile - Amendment*
IETF VPLS OAM	*L2VPN OAM Requirements and Framework*
ITU-T	*ITU-T Y.1731 OAM Mechanisms for Ethernet-Based Networks*

Technical Assistance

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

Feature Information for Using Ethernet Operations Administration and Maintenance

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

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

Table 1: Feature Information for Using Ethernet Operations, Administration, and Maintenance

Feature Name	Releases	Feature Information
Ethernet Operations, Administration, and Maintenance	12.4(15)T	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. The OAM features covered by this protocol are Discovery, Link Monitoring, Remote Fault Detection, Remote Loopback, and Cisco Proprietary Extensions. The Ethernet Operations, Administration, and Maintenance feature was integrated into Cisco IOS Release 12.4(15)T. The following commands were introduced or modified: **clear ethernet oam statistics, debug ethernet oam, ethernet oam, ethernet oam link-monitor frame, ethernet oam link-monitor frame-period, ethernet oam link-monitor frame-seconds, ethernet oam link-monitor high-threshold action, ethernet oam link-monitor on, ethernet oam link-monitor receive-crc, ethernet oam link-monitor supported, ethernet oam link-monitor symbol-period, ethernet oam link-monitor transmit-crc, ethernet oam remote-loopback, ethernet oam remote-loopback (interface), show ethernet oam discovery, show ethernet oam statistics, show ethernet oam status, show ethernet oam summary, source template (eoam), template (eoam)**.

CHAPTER 3

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.

Restrictions for Configuring Ethernet CFM in a Service Provider Network

- CFM loopback messages will not be confined within a maintenance domain according to their maintenance level. The impact of not having CFM loopback messages confined to their maintenance levels occurs at these levels:
    - Architecture—CFM layering is violated for loopback messages.
    - Deployment—A user may potentially misconfigure a network and have loopback messages succeed.
    - Security—A malicious device that recognizes devices' MAC addresses and levels may potentially explore a network topology that should be transparent.
- CFM is not fully supported on a Multiprotocol Label Switching (MPLS) provider edge (PE) device. There is no interaction between CFM and an Ethernet over MPLS (EoMPLS) pseudowire.
- CFM configuration is not supported on an EtherChannel in FastEthernet Channel (FEC) mode.
- QinQ encapsulation is not supported on the Cisco ASR 1000 Series Aggregation Services Router for CFM for routed subinterfaces.

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.

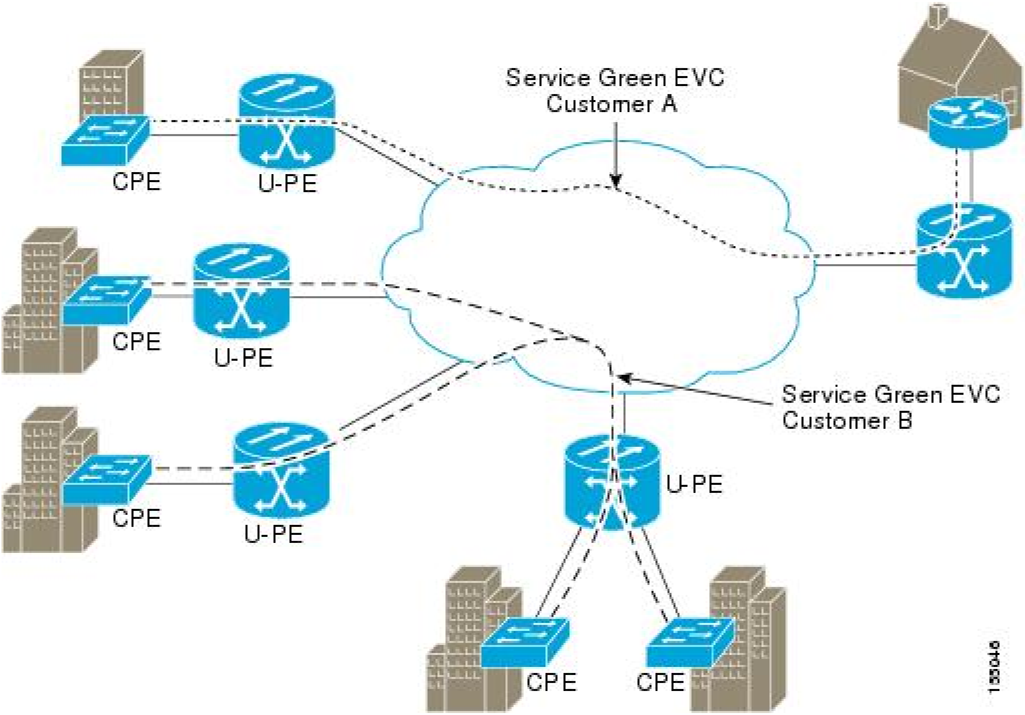
Benefits of Ethernet CFM

- 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

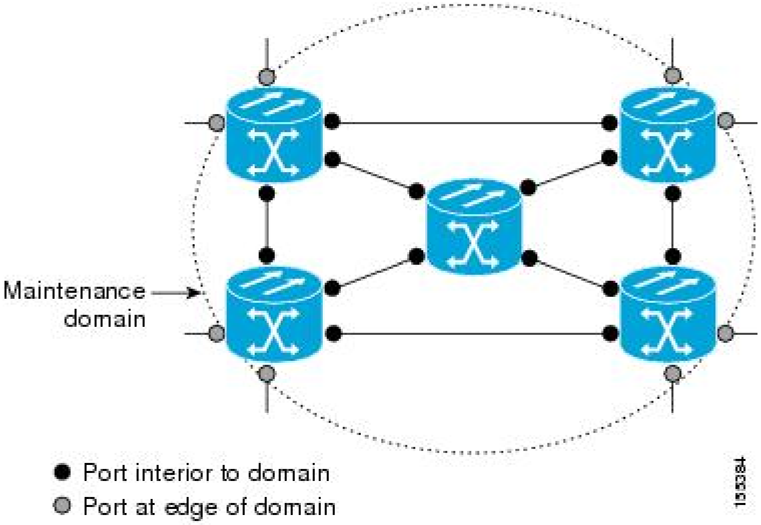
Customer Service Instance

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.

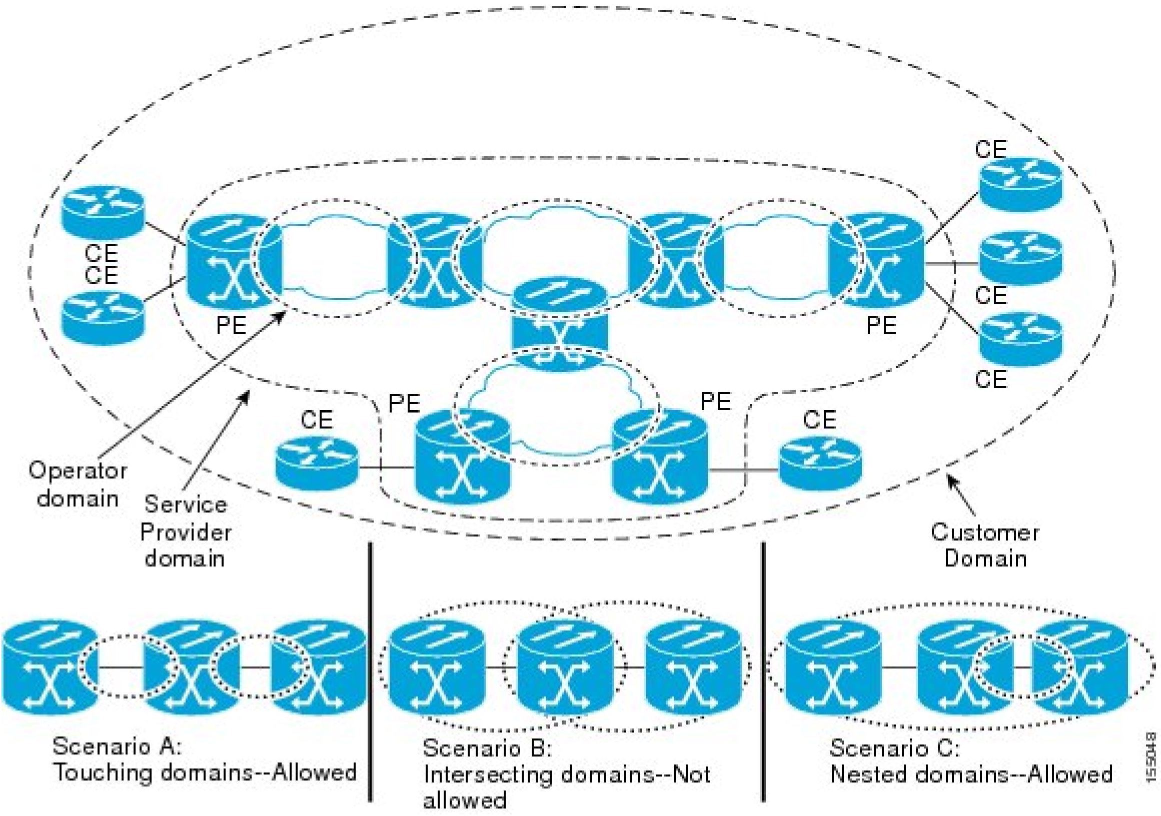


A unique maintenance level in the range of 0 to 7 is assigned to each domain by a network administrator. Levels and domain names are useful for defining the hierarchical relationship that exists among domains. The hierarchical relationship of domains parallels the structure of customer, service provider, 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 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 its level or 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 for Port Channels

Outward facing means that the MEP communicates through the wire. Outward facing MEPs can be configured on port channels (using cross connect functionality). A MIP configuration at a level higher than the level of the outward facing MEP is not required.

Outward facing MEPs on port channels use the Bridge-Brain MAC address of the first member link. When port channel members change, the identities of outward facing MEPs do not have to change.

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:

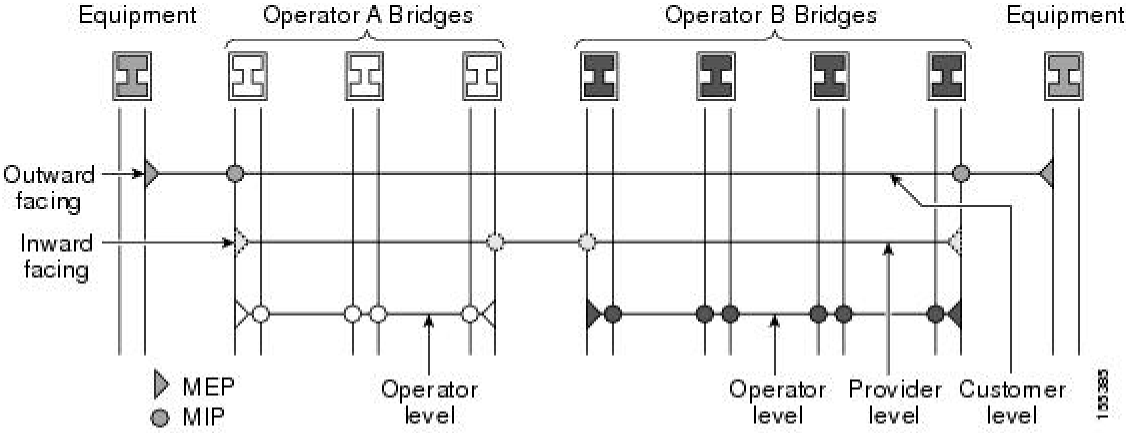
- Per maintenance domain (level) and for all S-VLANs enabled or allowed on a port.
- Internal to a domain, not at the boundary.
- CFM frames received from MEPs and other MIPs are cataloged and forwarded, using both the wire and the relay function.

- 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 multicast 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.

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 30.
- Contain a configurable hold-time value to indicate to the receiver the validity of the message. The default is 2.5 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.

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.

Cross-Check Function

The cross-check function is a timer-driven post-provisioning service verification between dynamically discovered MEPs (via CCMs) and expected MEPs (via configuration) for a service. The cross-check function verifies that all endpoints of a multipoint or point-to-point service are operational. The function supports notifications when the service is operational; otherwise it provides alarms and notifications for unexpected endpoints or missing endpoints.

The cross-check function is performed one time. You must initiate the cross-check function from the CLI every time you want a service verification.

SNMP Traps

The support provided by the Cisco software implementation of 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.

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.

HA Features Supported by CFM

In access and service provider networks using Ethernet technology, High Availability (H)A is a requirement, especially on Ethernet OAM components that manage EVC connectivity. End-to-end connectivity status information is critical and must be maintained on a hot standby Route Switch Processor (RSP).

Note A hot standby Route Switch Processor (RSP) has the same software image as the active RSP and supports synchronization of protocol and application state information between RSPs 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 Connectivity Fault Management (CFM) and 802.3ah. This status information is used to either stop traffic or switch to backup paths when an EVC is down.

Every transaction involves either accessing or updating data among various databases. If the database is synchronized across active and standby modules, the modules are transparent to clients.

The Cisco infrastructure provides various component application program interfaces (APIs) that help to maintain a hot standby RSP. Metro Ethernet HA clients HA/ISSU, CFM HA/ISSU, and 802.3ah HA/ISSU interact with these components, update the database, and trigger necessary events to other components.

Benefits of CFM HA

- Elimination of network downtime for Cisco software image upgrades, allowing for faster upgrades.
- Elimination of resource scheduling challenges associated with planned outages and late night maintenance windows.

- Accelerated deployment of new services and applications and facilitation of faster implementation of new features.
- Reduced operating costs due to outages while delivering higher service levels.
- CFM updates its databases and controls its own HA messaging and versioning, and this control facilitates maintenance.

CFM HA in a Metro Ethernet Network

A standalone Connectivity Fault Management (CFM) implementation does not have explicit high availability (HA) requirements. When CFM is implemented on a customer edge (CE) or provider edge (PE), CFM must maintain the Ethernet virtual circuit (EVC) state, which requires HA because the EVC state is critical in maintaining end-to-end connectivity. CFM configures the platform with maintenance level, domain, and maintenance point, learns the remote maintenance point information, and maps it to the appropriate EVC. CFM then aggregates data received from all remote ports; consequently HA requirements vary for CE and PE.

The CE receives the EVC ID, associated customer VLANs, UNI information, EVC state, and remote UNI ID and state from the MEN. The CE relies on the EVC state to send or stop traffic to the MEN.

The PE has EVC configuration and associated customer VLAN information and derives the EVC state and remote UNI from CFM.

Note PEs and CEs running 802.3ah OAM must maintain the port state so peers are not affected by a switchover. This information is also sent to remote nodes in CFM CC messages.

NSF SSO Support in CFM 802.1ag 1.0d

The redundancy configurations Stateful Switchover (SSO) and Nonstop Forwarding (NSF) are both supported in Ethernet Connectivity Fault Management (CFM) and are automatically enabled. A switchover from an active to a standby Route Switch Processor (RSP) occurs when the active RSP fails, is removed from the networking device, or is manually taken down for maintenance. NSF interoperates with the SSO feature to minimize network downtime following a switchover. The primary function of Cisco NSF is to continue forwarding IP packets following an RSP switchover.

For detailed information about SSO, see the “Configuring Stateful Switchover” module of the *High Availability Configuration Guide*. For detailed information about the NSF feature, see the “Configuring Cisco Nonstop Forwarding” module of the *High Availability Configuration Guide*.

ISSU Support in CFM 802.1ag 1.0d

In Service Upgrades (ISSUs) allow you to perform a Cisco software upgrade or downgrade without disrupting packet flow. Connectivity Fault Management (CFM) performs a bulk update and a runtime update of the continuity check database to the standby Route Switch Processor (RSP), including adding, deleting, or updating a row. This checkpoint data requires ISSU capability to transform messages from one release to another. All the components that perform active RSP to standby RSP updates using messages require ISSU support.

ISSU is automatically enabled in CFM and 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 “Performing an In Service Software Upgrade ” module of the *High Availability Configuration Guide*.

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.

SUMMARY STEPS

1. Determine operator level MIPs.
2. Determine operator level MEPs.
3. Determine service provider MIPs.
4. Determine service provider MEPs.
5. Determine customer MIPs.
6. Determine customer MEPs.

DETAILED STEPS

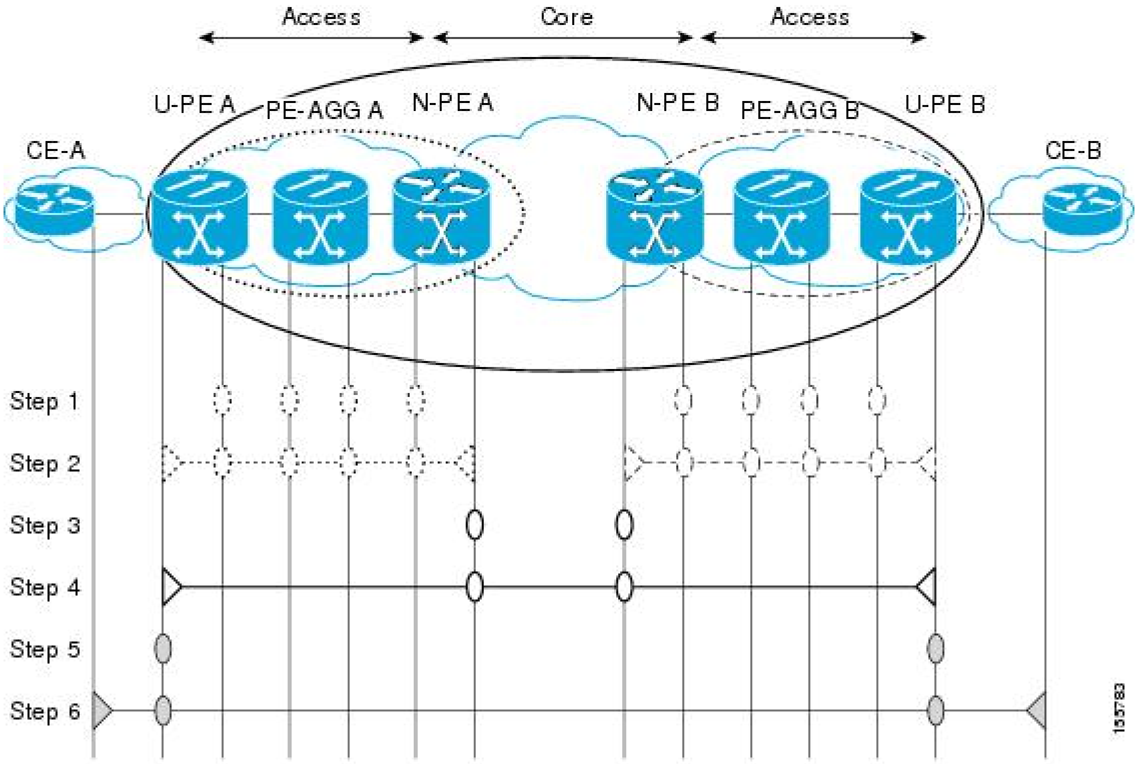
	Command or Action	Purpose
Step 1	Determine operator level MIPs.	Follow these steps: • Starting at lowest operator level domain, assign a MIP at every interface internal to the operator network to be visible to CFM.

	Command or Action	Purpose
		• 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.

	Command or Action	Purpose
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.

Examples

The figure below shows an example of a network with a service provider and two operators, A and B. Three domains are to be established to map to each operator and the service provider. In this example, for simplicity we assume that the network uses Ethernet transport end to end. CFM, however, can be used with other transports.



What to Do Next

After you have defined the Ethernet CFM domains, configure Ethernet CFM functionality by first provisioning the network and then provisioning service.

Configuring Ethernet CFM

Configuring Ethernet CFM consists of the following tasks:

Provisioning the Network

Provisioning the Network on the CE-A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
15. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]
16. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**ethernet cfm traceroute cache size** *entries* **Example:**	Sets the maximum size for the CFM traceroute cache table.

	Command or Action	Purpose
	`Device(config)# ethernet cfm traceroute cache size 200`	
Step 13	**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 14	**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 15	**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 16	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Provisioning the Network on the U-PE A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*

14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mip level** { *level* }
19. **exit**
20. **exit**
21. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
22. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
Step 7	**exit** **Example:**	Returns to Ethernet connectivity fault management configuration mode.

	Command or Action	Purpose
	`Device(config-ecfm-srv)# exit`	
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose

Step 18	**cfm mip level** { *level* }
	Example:
	`Device(config-if-srv)#cfm mip level 4`
		Creates a MIP and sets the maintenance level number.

Step 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 21	**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 mep-up, mep-down, config, loop, and cross-connect events.
Step 22	**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 mep-unknown, mep-missing, and service-up continuity check events in relation to the cross-check operation between statically configured MEPs and those learned via CCMs.

Step 23	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Provisioning the Network on the PE-AGG A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*

9. **exit**
10. **ethernet cfm global**
11. **interface** *type number*
12. **service instance** *id* **ethernet** [*evc-name*]
13. **encapsulation** *encapsulation-type*
14. **bridge-domain** *bridge-id*
15. **cfm mip level** *level*
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
Step 7	**exit** **Example:** `Device(config-ecfm-srv)# exit`	Returns to Ethernet connectivity fault management configuration mode.

	Command or Action	Purpose
Step 8	**mep archive-hold-time** *minutes* **Example:** `Device(config-ecfm)# mep archive-hold-time 65`	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 the CLI to global configuration mode.
Step 10	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 11	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 12	**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 13	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 14	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 15	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.

Step 16	**end**
	Example:
	`Device(config-if)# end`
		Returns to privileged EXEC mode.

Provisioning the Network on the N-PE A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*

4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **ethernet cfm global**
9. **ethernet cfm traceroute cache**
10. **ethernet cfm traceroute cache size** *entries*
11. **ethernet cfm traceroute cache hold-time** *minutes*
12. **interface** *type number*
13. **service instance** *id* **ethernet** [*evc-name*]
14. **encapsulation** *encapsulation-type*
15. **bridge-domain** *bridge-id*
16. **cfm mip level** *level*
17. **exit**
18. **exit**
19. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
20. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]
21. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures the transmission of continuity check messages (CCMs).

	Command or Action	Purpose
Step 6	**continuity-check** [**interval** *cc-interval*] **Example:** `Device(config-ecfm-srv)# continuity-check interval 10s`	Configures the per-service parameters and sets the interval at which CCMs are transmitted.
Step 7	**exit** **Example:** `Device(config-ecfm-srv)# exit`	Returns to Ethernet connectivity fault management configuration mode.
Step 8	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 9	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 10	**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 11	**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 12	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 13	**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 14	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 15	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
Step 16	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.
Step 17	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 18	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 19	**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 mep-up, mep-down, config, loop, and cross-connect events.
Step 20	**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 mep-unknown, mep-missing, and service-up continuity check events in relation to the cross-check operation between statically configured MEPs and those learned via CCMs.

Step 21	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Provisioning the Network on the CE-B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*

9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
15. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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.

	Command or Action	Purpose
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**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 mep-up, mep-down, config, loop, and cross-connect events.
Step 15	**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 mep-unknown, mep-missing, and service-up continuity check events in relation to the cross-check operation between statically configured MEPs and those learned via CCMs.

Step 16	**end**
	Example:
	`Device(config)# end#`
		Returns to privileged EXEC mode.

Provisioning the Network on the U-PE B

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mip level** *level*
19. **exit**
20. **exit**
21. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
22. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:**	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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.

	Command or Action	Purpose
Step 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 18	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.
Step 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 21	**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 mep-up, mep-down, config, loop, and cross-connect events.
Step 22	**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 mep-unknown, mep-missing, and service-up continuity check events in relation to the cross-check operation between statically configured MEPs and those learned via CCMs.

Step 23	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Provisioning the Network on the PE-AGG B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**

6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **interface** *type number*
12. **service instance** *id* **ethernet** [*evc-name*]
13. **encapsulation** *encapsulation-type*
14. **bridge-domain** *bridge-id*
15. **cfm mip level** *level*
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
Step 7	**exit** **Example:** `Device(config-ecfm-srv)# exit`	Returns to Ethernet connectivity fault management configuration mode.

	Command or Action	Purpose
Step 8	**mep archive-hold-time** *minutes* **Example:** `Device(config-ecfm)# mep archive-hold-time 65`	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	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 12	**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 13	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 14	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 15	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.

Step 16	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning the Network on the N-PE B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]

7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mip level** *level*
19. **exit**
20. **exit**
21. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
22. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures the transmission of continuity check messages (CCMs).
Step 6	**continuity-check** [**interval** *cc-interval*] **Example:**	Configures the per-service parameters and sets the interval at which CCMs are transmitted.

	Command or Action	Purpose
	`Device(config-ecfm-srv)# continuity-check interval 10s`	
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
Step 18	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.
Step 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 21	**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 mep-up, mep-down, config, loop, and cross-connect events.
Step 22	**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 mep-unknown, mep-missing, and service-up continuity check events in relation to the cross-check operation between statically configured MEPs and those learned via CCMs.

Step 23	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Provisioning Service

Provisioning Service on the CE-A

Perform this task to set up service for Ethernet CFM. Optionally, when this task is completed, you may configure and enable the cross-check function. To perform this optional task, see “Configuring and Enabling Cross-Checking for an Inward Facing MEP on the U PE-A”.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]

7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mep domain** *domain-name* **mpid** *id*
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.

	Command or Action	Purpose
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
Step 18	**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 19	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the U-PE A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mep domain** *domain-name* **mpid** *id*
19. **exit**
20. **exit**
21. **interface** *type number*
22. **service instance** *id* **ethernet** [*evc-name*]
23. **encapsulation** *encapsulation-type*
24. **bridge-domain** *bridge-id*
25. **cfm mip level** *level*
26. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.

	Command or Action	Purpose
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 18	**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 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 21	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 22	**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.

	Command or Action	Purpose
Step 23	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 24	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 25	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.

Step 26	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the PE-AGG A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **interface** *type number*
12. **service instance** *id* **ethernet** [*evc-name*]
13. **encapsulation** *encapsulation-type*
14. **bridge-domain** *bridge-id*
15. **cfm mip level** *level*
16. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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 65`	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	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 12	**service instance** *id* **ethernet** [*evc-name*] **Example:**	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.

	Command or Action	Purpose
	`Device(config-if)# service instance 333 ethernet evc1`	
Step 13	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 14	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 15	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.

Step 16	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the N-PE A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mip level** *level*
19. **exit**
20. **exit**
21. **interface** *type number*
22. **service instance** *id* **ethernet** [*evc-name*]
23. **encapsulation** *encapsulation-type*

24. **bridge-domain** *bridge-id*
25. **cfm mep domain** *domain-name* **mpid** *id*
26. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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.

	Command or Action	Purpose
Step 10	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 11	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 18	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.
Step 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 21	**interface** *type number* **Example:**	Specifies an interface.
Step 22	**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 23	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 24	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 25	**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 26	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the CE-B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mep domain** *domain-name* **mpid** *id*

19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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:**	Enables CFM processing globally on the device.

	Command or Action	Purpose
	`Device(config)# ethernet cfm global`	
Step 11	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 18	**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 19	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the U-PE B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**

5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mip level** *level*
19. **exit**
20. **exit**
21. **interface** *type number*
22. **service instance** *id* **ethernet** [*evc-name*]
23. **encapsulation** *encapsulation-type*
24. **bridge-domain** *bridge-id*
25. **cfm mep domain** *domain-name* **mpid** *id*
26. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.

	Command or Action	Purpose
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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.

	Command or Action	Purpose
Step 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 18	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.
Step 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 21	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 22	**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 23	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 24	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 25	**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 26	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the PE-AGG B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **interface** *type number*
12. **service instance** *id* **ethernet** [*evc-name*]
13. **encapsulation** *encapsulation-type*
14. **bridge-domain** *bridge-id*
15. **cfm mip level** *level*
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures the transmission of continuity check messages (CCMs).

	Command or Action	Purpose
Step 6	**continuity-check** [**interval** *cc-interval*] **Example:** `Device(config-ecfm-srv)# continuity-check interval 10s`	Configures the per-service parameters and sets the interval at which CCMs are transmitted.
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 65`	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	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 12	**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 13	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 14	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 15	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.

Step 16	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Provisioning Service on the N-PE B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **continuity-check**
6. **continuity-check** [**interval** *cc-interval*]
7. **exit**
8. **mep archive-hold-time** *minutes*
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm traceroute cache**
12. **ethernet cfm traceroute cache size** *entries*
13. **ethernet cfm traceroute cache hold-time** *minutes*
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-name*]
16. **encapsulation** *encapsulation-type*
17. **bridge-domain** *bridge-id*
18. **cfm mip level** *level*
19. **exit**
20. **exit**
21. **interface** *type number*
22. **service instance** *id* **ethernet** [*evc-name*]
23. **encapsulation** *encapsulation-type*
24. **bridge-domain** *bridge-id*
25. **cfm mep domain** *domain-name* **mpid** *id*
26. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.

	Command or Action	Purpose
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.
Step 5	**continuity-check** **Example:** `Device(config-ecfm-srv)# continuity-check`	Configures 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 per-service parameters and sets the interval at which CCMs are transmitted.
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	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 12	**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 13	**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 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**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 16	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 17	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 18	**cfm mip level** *level* **Example:** `Device(config-if-srv)#cfm mip level 4`	Creates a MIP and sets the maintenance level number.
Step 19	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 21	**interface** *type number* **Example:**	Specifies an interface.
Step 22	**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 23	**encapsulation** *encapsulation-type* **Example:**	Sets the encapsulation method used by the interface.
Step 24	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 25	**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.

	Command or Action	Purpose

Step 26	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Configuring and Enabling the Cross-Check Function

Configuring and Enabling Cross-Checking for an Inward Facing MEP on the U PE-A

Perform this task to configure and enable cross-checking for an inward facing MEP. This task requires you to configure and enable cross-checking on two devices. This task is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*]
5. **exit**
6. **ethernet cfm mep crosscheck start-delay** *delay*
7. **exit**
8. **ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain ServiceProvider level 4`	Defines a CFM domain at a specified level and enters Ethernet CFM configuration mode.
Step 4	**mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*] **Example:** `Device(config-ether-cfm)# mep crosscheck mpid 402 vlan 100`	Statically defines a remote MEP on a specified VLAN within the domain.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-ether-cfm)# exit#`	Returns to global configuration mode.
Step 6	**ethernet cfm mep crosscheck start-delay** *delay* **Example:** `Device(config)# ethernet cfm mep crosscheck start-delay 60`	Configures the maximum amount of time that the device waits for remote MEPs to come up before the cross-check operation is started
Step 7	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 8	**ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]} **Example:** `Device# ethernet cfm mep crosscheck enable level 4 vlan 100`	Enables cross-checking between remote MEPs in the domain and MEPs learned through CCMs.

Example

The following example configures cross-checking on an inward facing MEP (U-PE A):

```
U-PE A
ethernet cfm domain ServiceProvider level 4
mep crosscheck mpid 402 vlan 100
!
ethernet cfm mep crosscheck start-delay 60
```

The following example enables cross-checking on an inward facing MEP (U-PE A):

```
U-PE A
U-PEA# ethernet cfm mep crosscheck enable level 4 vlan 100
```

Configuring and Enabling Cross-Checking for an Inward Facing MEP on the U PE-B

Perform this task to configure and enable cross-checking for an inward facing MEP. This task requires you to configure and enable cross-checking on two devices. This task is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*]
5. **exit**
6. **ethernet cfm mep crosscheck start-delay** *delay*
7. **exit**

8. **ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain ServiceProvider level 4`	Defines a CFM domain at a specified level and enters Ethernet CFM configuration mode.
Step 4	**mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*] **Example:** `Device(config-ether-cfm)# mep crosscheck mpid 401 vlan 100`	Statically defines a remote MEP on a specified VLAN within the domain.
Step 5	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 6	**ethernet cfm mep crosscheck start-delay** *delay* **Example:** `Device(config)# ethernet cfm mep crosscheck start-delay 60`	Configures the maximum amount of time that the device waits for remote MEPs to come up before the cross-check operation is started.
Step 7	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 8	**ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]} **Example:** `Device# ethernet cfm mep crosscheck enable level 4 vlan 100`	Enables cross-checking between MEPs.

Example

The following example configures cross-checking on an inward facing MEP (U-PE B)

```
U-PE B
ethernet cfm domain ServiceProvider level 4
mep crosscheck mpid 401 vlan 100
!
ethernet cfm mep crosscheck start-delay 60
```

The following example enables cross-checking on an inward facing MEP (U-PE B)

```
U-PE B
U-PEB# ethernet cfm mep crosscheck enable level 4 vlan 100
```

Configuring and Enabling Cross-Checking for an Outward Facing MEP on the CE-A

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**]
4. **mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*]
5. **exit**
6. **ethernet cfm mep crosscheck start-delay** *delay*
7. **exit**
8. **ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**] **Example:** `Device(config)# ethernet cfm domain Customer level 7 direction outward`	Defines a CFM domain at a specified level and enters Ethernet CFM configuration mode.
Step 4	**mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*] **Example:**	Statically defines a remote MEP with a specified ID, VLAN, and domain.

	Command or Action	Purpose
	`Device(config-ether-cfm)# mep crosscheck mpid 702 vlan 100`	
Step 5	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 6	**ethernet cfm mep crosscheck start-delay** *delay* **Example:** `Device(config)# ethernet cfm mep crosscheck start-delay 60`	Configures the maximum amount of time that the device waits for remote MEPs to come up before the cross-check operation is started.
Step 7	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 8	**ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]} **Example:** `Device# ethernet cfm mep crosscheck enable level 7 vlan 100`	Enables cross-checking between MEPs.

Configuring and Enabling Cross-Checking for an Outward Facing MEP on the CE-B

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**]
4. **mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*]
5. **exit**
6. **ethernet cfm mep crosscheck start-delay** *delay*
7. **exit**
8. **ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]}

DETAILED STEPS

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

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**] **Example:** `Device(config)# ethernet cfm domain Customer level 7 direction outward`	Defines an outward CFM domain at a specified level and enters Ethernet CFM configuration mode.
Step 4	**mep crosscheck mpid** *id* **vlan** *vlan-id* [**mac** *mac-address*] **Example:** `Device(config-ether-cfm)# mep crosscheck mpid 401 vlan 100`	Statically defines a remote MEP on a VLAN within a specified domain.
Step 5	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 6	**ethernet cfm mep crosscheck start-delay** *delay* **Example:** `Device(config)# ethernet cfm mep crosscheck start-delay 60`	Configures the maximum amount of time that the device waits for remote MEPs to come up before the cross-check operation is started.
Step 7	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 8	**ethernet cfm mep crosscheck** {**enable** | **disable**} **level** {*level-id* | *level-id-level-id* [,*level-id-level-id*]} **vlan** {*vlan-id* | **any** | *vlan-id-vlan-id* [,*vlan-id-vlan-id*]} **Example:** `Device# ethernet cfm mep crosscheck enable level 7 vlan 100`	Enables cross-checking between MEPs.

Configuring CFM over Bridge Domains

Perform this task to configure Ethernet CFM over bridge domains. This task is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id* **direction outward**
4. **service** *csi-id* **evc** *evc-name*
5. **exit**
6. **ethernet cfm domain** *domain-name* **level** *level-id*
7. **exit**

8. **ethernet cfm domain** *domain-name* **level** *level-id*
9. **service** *csi-id* **evc** *evc-name*
10. **mep crosscheck mpid** *id* **evc** *evc-name* **mac** *mac-address*
11. **exit**
12. **ethernet evc** *evc-name*
13. **exit**
14. **interface** *type number*
15. **no ip address**
16. **service instance** *id* **ethernet** *evc-id*
17. **encapsulation dot1q** *vlan-id*
18. **bridge-domain** *bridge-id*
19. **cfm mep domain** *domain-name* **mpid** *mpid-value*
20. **end**
21. **configure terminal**
22. **interface** *type name*
23. **no ip address**
24. **ethernet cfm mip level** *level-id*
25. **service instance** *id* **ethernet** *evc-id*
26. **encapsulation dot1q** *vlan-id*
27. **bridge-domain** *bridge-id*
28. **cfm mep domain** *domain-name* **mpid** *mpid-value*
29. **end**
30. **configure terminal**
31. **ethernet cfm cc enable level** *level-id* **evc** *evc-name*
32. **ethernet cfm cc level any evc** *evc-name* **interval** *seconds* **loss-threshold** *num-msgs*
33. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **direction outward** **Example:**	Defines a CFM maintenance domain at a particular level and enters Ethernet CFM configuration mode.

	Command or Action	Purpose
	`Device(config)# ethernet cfm domain CUSTOMER level 7 direction outward`	
Step 4	**service** *csi-id* **evc** *evc-name* **Example:** `Device(config-ether-cfm)# service customer_100 evc evc_100`	Sets a universally unique ID for a CSI within a maintenance domain.
Step 5	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 6	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain MIP level 7`	Defines a CFM maintenance domain at a particular level and enters Ethernet CFM configuration mode.
Step 7	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 8	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain PROVIDER level 4`	Defines a CFM maintenance domain at a particular level and enters Ethernet CFM configuration mode.
Step 9	**service** *csi-id* **evc** *evc-name* **Example:** `Device(config-ether-cfm)# service provider_1 evc evc_100`	Sets a universally unique ID for a CSI within a maintenance domain.
Step 10	**mep crosscheck mpid** *id* **evc** *evc-name* **mac** *mac-address* **Example:** `Device(config-ether-cfm)# mep crosscheck mpid 200 evc evc_100 mac 1010.1010.1010`	Statically defines a remote MEP within a maintenance domain.
Step 11	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 12	**ethernet evc** *evc-name* **Example:** `Device(config)# ethernet evc evc_100`	Defines an EVC and enters EVC configuration mode.
Step 13	**exit** **Example:** `Device(config-evc)# exit`	Returns to global configuration mode.
Step 14	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.
Step 15	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 16	**service instance** *id* **ethernet** *evc-id* **Example:** `Device(config-if)# service instance 100 ethernet evc_100`	Specifies an Ethernet service instance on an interface and enters service instance configuration mode.
Step 17	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to map 802.1Q frames on an ingress interface to the appropriate service instance.
Step 18	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Establishes a bridge domain.
Step 19	**cfm mep domain** *domain-name* **mpid** *mpid-value* **Example:** `Device(config-if-srv)# cfm mep domain CUSTOMER mpid 1001`	Configures a MEP for a domain.

Step 20	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Step 21	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 22	**interface** *type name* **Example:**	Specifies an interface and enters interface configuration mode.
Step 23	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 24	**ethernet cfm mip level** *level-id* **Example:** `Device(config-if)# ethernet cfm mip level 7`	Provisions a MIP at a specified maintenance level on an interface.
Step 25	**service instance** *id* **ethernet** *evc-id* **Example:** `Device(config-if)# service instance 100 ethernet evc_100`	Configures an Ethernet service instance on an interface and enters service instance configuration mode.
Step 26	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to map 802.1Q frames on an ingress interface to the appropriate service instance.
Step 27	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Establishes a bridge domain.
Step 28	**cfm mep domain** *domain-name* **mpid** *mpid-value* **Example:** `Device(config-if-srv)# cfm mep domain PROVIDER inward mpid 201`	Configures a MEP for a domain.

Step 29	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to privileged EXEC mode.

Step 30	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 31	**ethernet cfm cc enable level** *level-id* **evc** *evc-name* **Example:** `Device(config)# ethernet cfm cc enable level 0-7 evc evc_100`	Globally enables transmission of CCMs.
Step 32	**ethernet cfm cc level any evc** *evc-name* **interval** *seconds* **loss-threshold** *num-msgs* **Example:** `Device(config)# ethernet cfm cc level any evc evc_100 interval 100 loss-threshold 2`	Sets the parameters for CCMs.

Step 33	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

What to do next

Note When configuring CFM over bridge domains where the bridge-domain ID matches the vlan ID service, you must configure the vlan service and the EVC service with the same service name. The bridge-domain is associated with the EVC service. The vlan and the bridge-domain represent the same broadcast domain.

Configuring CFM Over Port Channels

Configuring UP MEP over Port Channel in L2VPN

Perform this task to configure up Maintenance End Point (MEP) over port channel in Layer 2 VPN (L2VPN). This task shows Provider Edge 1 and 2 configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm ieee**
4. **ethernet cfm global**
5. **ethernet cfm domain** *domain-name* **level** *level-id*
6. **service** *csi-id* **evc** *evc-name*
7. **continuity-check** [**inteval** *time*]
8. **exit**
9. **ethernet evc** *evc-name*
10. **pseudowire-class** *pw-class-name*
11. **encapsulation mpls**

12. **exit**
13. **interface** *type number*
14. **service instance** *id* **ethernet** *evc-id*
15. **encapsulation dot1q** *vlan-id*
16. **rewrite ingress tag pop 1 symmetric**
17. **xconnect** *peer-ip-addressvc-id* **pw-class** *pw-class-name*
18. **cfm mep domain** *domain-name* **mpid** *mpid-value*
19. *exit*
20. **ethernet cfm ieee**
21. **ethernet cfm global**
22. **ethernet cfm domain** *domain-name* **level** *level-id*
23. **service** *csi-id* **evc** *evc-name*
24. **continuity-check** [**inteval** *ime*]
25. **exit**
26. **ethernet evc** *evc-name*
27. **pseudowire-class** *pw-class-name*
28. **encapsulation mpls**
29. **exit**
30. **interface** *type number*
31. **service instance** *id* **ethernet** *evc-id*
32. **encapsulation dot1q** *vlan-id*
33. **rewrite ingress tag pop 1 symmetric**
34. **xconnect** *peer-ip-addressvc-id* **pw-class** *pw-class-name*
35. **cfm mep domain** *domain-name* **mpid** *mpid-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 4	**ethernet cfm global** **Example:**	Enables Ethernet connectivity fault management (CFM) globally on a device.

	Command or Action	Purpose
	`Device(config)# ethernet cfm global`	
Step 5	**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 level and enters Ethernet CFM configuration mode.
Step 6	**service** *csi-id* **evc** *evc-name* **Example:** `Device(config-ether-cfm)# service customer100 evc evc100`	Sets a universally unique ID for a customer service instance (CSI) within a maintenance domain.
Step 7	**continuity-check** [**inteval** *time*] **Example:** `Device(config-ether-cfm)# continuity-check interval 1s`	Enables the transmission of continuity check messages (CCMs)
Step 8	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 9	**ethernet evc** *evc-name* **Example:** `Device(config)# ethernet evc evc100`	Enables Ethernet Virtual Circuit (EVC).
Step 10	**pseudowire-class** *pw-class-name* **Example:** `Device(config-evc)# pseudowire-class vlan-xconnect`	Specifies the name of a Layer 2 pseudowire class and enter pseudowire class configuration mode.
Step 11	**encapsulation mpls** **Example:** `Device(config-pw)# encapsulation mpls`	Specifies an encapsulation type for tunneling Layer 2 traffic over a pseudowire
Step 12	**exit** **Example:** `Device (config-ecfm-srv)# exit`	Exits Ethernet CFM configuration mode and returns to global configuration mode.
Step 13	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface port-channel 10`	
Step 14	**service instance** *id* **ethernet** *evc-id* **Example:** `Device(config-if)# service instance 100 ethernet evc100`	Specifies an Ethernet service instance on an interface and enters service instance configuration mode.
Step 15	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to map 802.1Q frames on an ingress interface to the appropriate service instance.
Step 16	**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 ingressing a service instance.
Step 17	**xconnect** *peer-ip-addressvc-id* **pw-class** *pw-class-name* **Example:** `Device(config-if-srv)#  xconnect 10.1.1.1 100 pw-class vlan-xconnect`	Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance.
Step 18	**cfm mep domain** *domain-name* **mpid** *mpid-value* **Example:** `Device(config-if-srv)# cfm mep domain CUSTOMER mpid 1111`	Configures a MEP for a domain.
Step 19	*exit* **Example:** `Device(config-if-srv)# exit`	Exits service instance configuration mode and enters global configuration mode. **Note** The configuration for Provider Edge Device 1 (PE1) ends here. Perform the next steps for PE2 configuration.
Step 20	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 21	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables Ethernet connectivity fault management (CFM) globally on a device.

	Command or Action	Purpose
Step 22	**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 level and enters Ethernet CFM configuration mode.
Step 23	**service** *csi-id* **evc** *evc-name* **Example:** `Device(config-ether-cfm)# service customer100 evc evc100`	Sets a universally unique ID for a customer service instance (CSI) within a maintenance domain.
Step 24	**continuity-check** [**inteval** *ime*] **Example:** `Device(config-ether-cfm)# continuity-check interval 1s`	Enables the transmission of continuity check messages (CCMs)
Step 25	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 26	**ethernet evc** *evc-name* **Example:** `Device(config)# ethernet evc evc_100`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 27	**pseudowire-class** *pw-class-name* **Example:** `Device(config-evc)# pseudowire-class vlan-xconnect`	Specifies the name of a Layer 2 pseudowire class and enter pseudowire class configuration mode.
Step 28	**encapsulation mpls** **Example:** `Device(config-pw)# encapsulation mpls`	Specifies an encapsulation type for tunneling Layer 2 traffic over a pseudowire
Step 29	**exit** **Example:** `Device (config-ecfm-srv)# exit`	Exits Ethernet CFM configuration mode and returns to global configuration mode.
Step 30	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet0/0/1`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 31	**service instance** *id* **ethernet** *evc-id* **Example:** `Device(config-if)# service instance 100 ethernet evc100`	Specifies an Ethernet service instance on an interface and enters service instance configuration mode.
Step 32	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to map 802.1Q frames on an ingress interface to the appropriate service instance.
Step 33	**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 ingressing a service instance.
Step 34	**xconnect** *peer-ip-addressvc-id* **pw-class** *pw-class-name* **Example:** `Device(config-if-srv)#  xconnect 10.1.1.2 100 pw-class vlan-xconnect`	Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance.
Step 35	**cfm mep domain** *domain-name* **mpid** *mpid-value* **Example:** `Device(config-if-srv)# cfm mep domain CUSTOMER mpid 2222`	Configures a MEP for a domain.

Configuring UP MEP over Port Channel in VPLS

Perform this task to configure up Maintenance End Point (MEP) over port channel in VPLS. This task shows configurations for Provider Edge (PE)1 and PE2 devices.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm ieee**
4. **ethernet cfm global**
5. **ethernet cfm domain** *domain-name* **level** *level-id*
6. **service** *csi-id* **evc** *evc-name*
7. **continuity-check** [**inteval** *time*]
8. **exit**
9. **ethernet evc** *evc-name*
10. **exit**
11. **l2vpn vfi context** *name*

12. **vpn id** *vpn-id*
13. **evc** *evc-name*
14. **member** *ip-address* **encapsulation mpls**
15. **exit**
16. **interface** *type number*
17. **service instance** *id* **ethernet** *evc-id*
18. **encapsulation dot1q** *vlan-id*
19. **rewrite ingress tag pop 1 symmetric**
20. **cfm mep domain** *domain-name* **mpid** *mpid-value*
21. **exit**
22. **bridge-domain** *bridge-id*
23. **member** *interface-type-number* **service-instance** *service-id*
24. **member** *interface-type-number*
25. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 4	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables Ethernet connectivity fault management (CFM) globally on a device.
Step 5	**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 level and enters Ethernet CFM configuration mode.
Step 6	**service** *csi-id* **evc** *evc-name* **Example:**	Sets a universally unique ID for a customer service instance (CSI) within a maintenance domain.

	Command or Action	Purpose
	`Device(config-ether-cfm)# service customer100 evc evc100`	
Step 7	**continuity-check** [**inteval** *time*] **Example:** `Device(config-ether-cfm)# continuity-check interval 1s`	Enables the transmission of continuity check messages (CCMs)
Step 8	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 9	**ethernet evc** *evc-name* **Example:** `Device(config)# ethernet evc evc100`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 10	**exit** **Example:** `Device(config-evc)# exit`	Returns to global configuration mode.
Step 11	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks and enters VFI configuration mode.
Step 12	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 1`	Updates a VPN ID on a Virtual Private LAN Services (VPLS) instance.
Step 13	**evc** *evc-name* **Example:** `Device(config-vfi)# evc evc100`	Configures an EVC on a VPLS instance.
Step 14	**member** *ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# member 10.1.1.1 encapsulation mpls`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection.
Step 15	**exit** **Example:**	Exits VFI configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Device (config-vfi)# exit`	
Step 16	**interface** *type number* **Example:** `Device(config)# interface port-channel 10`	Specifies an interface and enters interface configuration mode.
Step 17	**service instance** *id* **ethernet** *evc-id* **Example:** `Device(config-if)# service instance 100 ethernet evc100`	Specifies an Ethernet service instance on an interface and enters service instance configuration mode.
Step 18	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to map 802.1Q frames on an ingress interface to the appropriate service instance.
Step 19	**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 ingressing a service instance.
Step 20	**cfm mep domain** *domain-name* **mpid** *mpid-value* **Example:** `Device(config-if-srv)# cfm mep domain CUSTOMER mpid 1001`	Configures a MEP for a domain.
Step 21	**exit** **Example:** `Device (config-if-srv)# exit`	Exits service instance configuration mode and returns to global configuration mode.
Step 22	**bridge-domain** *bridge-id* **Example:** `Device(config)# bridge-domain 100`	Configures the components on a bridge domain and enters bridge-domain configuration mode.
Step 23	**member** *interface-type-number* **service-instance** *service-id* **Example:** `Device(config-bdomain)# member port-channel 10 service-instance 100`	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
Step 24	**member** *interface-type-number* **Example:** `Device(config-bdomain)# member vfi vpls1`	Binds a service instance to a bridge domain instance.
Step 25	**exit** **Example:** `Device (config-bdomain)# exit`	Exits bridge domain configuration mode and returns to global configuration mode.

Configuring Down MEP over Port Channel

Perform this task to configure down MEP over port channel.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm ieee**
4. **ethernet cfm global**
5. **ethernet cfm domain** *domain-name* **level** *level-id*
6. **service** *csi-id* **evc** *evc-name* s **vlan** *vlan-id* **direction down**
7. **continuity-check** [**inteval** *time*]
8. **exit**
9. **ethernet evc** *evc-name*
10. **exit**
11. **interface** *type number*
12. **service instance** *id* **ethernet** *evc-id*
13. **encapsulation dot1q** *vlan-id*
14. **bridge-domain** *bridge-id*
15. **cfm mep domain** *domain-name* **mpid** *mpid-value*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 4	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables Ethernet connectivity fault management (CFM) globally on a device.
Step 5	**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 level and enters Ethernet CFM configuration mode.
Step 6	**service** *csi-id* **evc** *evc-name* s **vlan** *vlan-id* **direction down** **Example:** `Device(config-ether-cfm)# service customer_100 evc evc_100 vlan 100 direction down`	Configures a maintenance association within a maintenance domain and enter Ethernet connectivity fault management (CFM) service configuration mode.
Step 7	**continuity-check** [**inteval** *time*] **Example:** `Device(config-ether-cfm)# continuity-check interval 1s`	Enables the transmission of continuity check messages (CCMs)
Step 8	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 9	**ethernet evc** *evc-name* **Example:** `Device(config)# ethernet evc evc_100`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 10	**exit** **Example:** `Device (config-ecfm-srv)# exit`	Exits Ethernet CFM configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface port-channel10`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 12	**service instance** *id* **ethernet** *evc-id* **Example:** `Device(config-if)# service instance 100 ethernet evc_100`	Specifies an Ethernet service instance on an interface and enters service instance configuration mode.
Step 13	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to map 802.1Q frames on an ingress interface to the appropriate service instance.
Step 14	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)#  bridge domain 100`	Configures the components on a bridge domain.
Step 15	**cfm mep domain** *domain-name* **mpid** *mpid-value* **Example:** `Device(config-if-srv)# cfm mep domain CUSTOMER mpid 2222`	Configures a MEP for a domain.

Configuring CFM Offload

Perform this task to configure Connectivity Fault Management (CFM) offload.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm ieee**
4. **ethernet cfm global**
5. **ethernet cfm domain** *domain-name* **level** *level-id*
6. **service** *csi-id* **evc** *evc-name*
7. **continuity-check** [**inteval** *time*]
8. **offload sampling** *sample*
9. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the Ethernet Connectivity Fault Management 802.1ag Standard (CFM IEEE) version of CFM.
Step 4	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables Ethernet connectivity fault management (CFM) globally on a device.
Step 5	**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 level and enters Ethernet CFM configuration mode.
Step 6	**service** *csi-id* **evc** *evc-name* **Example:** `Device(config-ether-cfm)# service customer100 evc evc100`	Sets a universally unique ID for a customer service instance (CSI) within a maintenance domain.
Step 7	**continuity-check** [**inteval** *time*] **Example:** `Device(config-ether-cfm)# continuity-check interval 3.3s`	Enables the transmission of continuity check messages (CCMs)
Step 8	**offload sampling** *sample* **Example:**	Configures offload sampling.

	Command or Action	Purpose
	`Device(config-ether-cfm)# offload sampling 1000`	**Note** ASR1000 routers can offload sessions with CCM interval of 100 milliseconds, 10 milliseconds, and 3.3 milliseconds. CCM session with 1 second interval does not get offloaded by default. To offload the CCM session with 1 second, configure the sampling rate (offload sampling). The CCM session with 10 minutes, 1 minute, and 10 seconds are not offloaded. The suggested offload sampling for each CCM interval is as follows: • 1s - 10 • 100ms - 100 • 10ms - 1000 • 3.3ms - 2000
Step 9	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.

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.

Configuring Ethernet OAM Interaction with CFM

For Ethernet OAM to function with CFM, you must configure an EVC and the OAM manager and associate the EVC with CFM. Additionally, you must use an inward facing MEP when you want interaction with the OAM manager.

Configuring the OAM Manager

Note If you configure, change, or remove a UNI service type, EVC, Ethernet service instance, or CE-VLAN configuration, all configurations are checked to ensure that UNI service types are matched with EVC configurations and Ethernet service instances are matched with CE-VLAN configurations. Configurations are rejected if the pairings do not match.

Perform this task to configure the OAM manager on a PE device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**]
4. **service** *csi-id* **vlan** *vlan-id*
5. **exit**
6. **ethernet evc** *evc-id*
7. **oam protocol** {**cfm svlan** *svlan-id* **domain** *domain-name* | **ldp**}
8. **exit**
9. Repeat Steps 3 through 8 to define other CFM domains that you want OAM manager to monitor.
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**] **Example:** `Device(config)# ethernet cfm domain cstmr1 level 3`	Defines a CFM domain, sets the domain level, and enters Ethernet CFM configuration mode.
Step 4	**service** *csi-id* **vlan** *vlan-id* **Example:** `Device(config-ether-cfm)# service csi2 vlan 10`	Defines a universally unique customer service instance (CSI) and VLAN ID within the maintenance domain.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-ether-cfm)# exit`	Returns to global configuration mode.
Step 6	**ethernet evc** *evc-id* **Example:** `Device(config)# ethernet evc 50`	Defines an EVC and enters EVC configuration mode.

Step 7	**oam protocol** {**cfm svlan** *svlan-id* **domain** *domain-name* | **ldp**}
	Example:
	`Device(config-evc)# oam protocol cfm svlan 10 domain cstmr1`
		Configures the EVC OAM protocol.

Step 8	**exit** **Example:** `Device(config-evc)# exit`	Returns to global configuration mode.
Step 9	Repeat Steps 3 through 8 to define other CFM domains that you want OAM manager to monitor.	—

Step 10	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Enabling Ethernet OAM

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 OAM on a device or on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	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 max-rate 50`	Enables Ethernet OAM on an interface.

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

Configuration Examples for Configuring Ethernet CFM in a Service Provider Network

Example: Provisioning a Network

This configuration example shows only CFM-related commands. All commands that are required to set up the data path and configure the VLANs on the device are not shown. However, it should be noted that CFM traffic will not flow into or out of the device if the VLANs are not properly configured.

```
CE-A
!
ethernet cfm domain Customer level 7
!!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!!
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
```

```
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up

U-PE A
!
ethernet cfm domain Customer level 7
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!

ethernet cfm mip level 1
!
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
PE-AGG A
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
!
ethernet cfm global
!

ethernet cfm mip level 1
!

ethernet cfm mip level 1
N-PE A
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!

ethernet cfm mip level 1
!
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
U-PE B
!
ethernet cfm domain Customer level 7
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
!
ethernet cfm domain OperatorB level 2
mep archive-hold-time 65
```

```
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!

ethernet cfm mip level 2
!
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
PE-AGG B
ethernet cfm domain OperatorB level 2
mep archive-hold-time 65
!
ethernet cfm global
!

ethernet cfm mip level 2
!

ethernet cfm mip level 2
N-PE B
!
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
!
ethernet cfm domain OperatorB level 2
mep archive-hold-time 65
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!

ethernet cfm mip level 2
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
CE-B
!
ethernet cfm domain Customer level 7
!!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!!
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

Example: Provisioning Service

This configuration example shows only CFM-related commands. All commands that are required to set up the data path and configure the VLANs on the device are not shown. However, it should be noted that CFM traffic will not flow into or out of the device if the VLANs are not properly configured.

```
CE-A
!
ethernet cfm domain Customer level 7
service Customer1 evc evc1 vlan 100

!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet0/0/2 / use an appropriate device-specific interface
ethernet cfm mep level 7 direction outward domain Customer1 mpid 701 vlan 100
!
ethernet cfm cc enable level 7 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
U-PE A
!
ethernet cfm domain Customer level 7
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
service MetroCustomer10pA evc evc1 vlan 100
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
 service MetroCustomer10pA evc evc1 vlan 100
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet0/0/2  /use an appropriate device-specific interface
ethernet cfm mip level 7
ethernet cfm mep level 4 mpid 401 vlan 100
ethernet cfm mep level 1 mpid 101 vlan 100
!
interface gigabitethernet0/0/2 /use an appropriate device-specific interface
ethernet cfm mip level 1
!
ethernet cfm cc enable level 4 vlan 100
ethernet cfm cc enable level 1 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
PE-AGG A
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
service MetroCustomer10pA evc evc1 vlan 100
!
ethernet cfm global
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 1
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 1
```

```
N-PE A
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
service MetroCustomer1 evc evc1 vlan 100
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
service MetroCustomer10pA evc evc1 vlan 100
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 1
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 4
ethernet cfm mep level 1 mpid 102 vlan 100
!
ethernet cfm cc enable level 1 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
U-PE B
!
ethernet cfm domain Customer level 7
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
service MetroCustomer1 evc evc1 vlan 100
!
ethernet cfm domain OperatorB level 2
mep archive-hold-time 65
service MetroCustomer10pB evc evc1 vlan 100
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 7
ethernet cfm mep level 4 mpid 402 vlan 100
ethernet cfm mep level 2 mpid 201 vlan 100
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 2
!
ethernet cfm cc enable level 4 vlan 100
ethernet cfm cc enable level 2 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
PE-AGG B
ethernet cfm domain OperatorB level 2
mep archive-hold-time 65
service MetroCustomer10pB evc evc1 vlan 100
!
ethernet cfm global
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 2
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 2
```

```
N-PE B
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
service MetroCustomer1 evc evc1 vlan 100
!
ethernet cfm domain OperatorB level 2
mep archive-hold-time 65
service MetroCustomer10pB evc evc1 vlan 100
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 2
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mip level 4
ethernet cfm mep level 2 mpid 202 vlan 100
!
ethernet cfm cc enable level 2 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
CE-B
!
ethernet cfm domain Customer level 7
service Customer1 vlan 100
!
ethernet cfm global
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet0/0/2 use an appropriate device-specific interface
ethernet cfm mep level 7 direction outward domain Customer1 mpid 702 vlan 100
!
ethernet cfm cc enable level 7 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
```

Glossary

CCM—continuity check message. A multicast CFM frame that a MEP transmits periodically to ensure continuity across the maintenance entities to which the transmitting MEP belongs, at the MA level on which the CCM is sent. No reply is sent in response to receiving a CCM.

EVC—Ethernet virtual connection. An association of two or more user-network interfaces.

fault alarm—An out-of-band signal, typically an SNMP notification, that notifies a system administrator of a connectivity failure.

inward-facing MEP—A MEP that resides in a bridge and transmits to and receives CFM messages from the direction of the bridge relay entity.

maintenance domain—The network or part of the network belonging to a single administration for which faults in connectivity are to be managed. The boundary of a maintenance domain is defined by a set of DSAPs, each of which may become a point of connectivity to a service instance.

maintenance domain name—The unique identifier of a domain that CFM is to protect against accidental concatenation of service instances.

MEP—maintenance endpoint. An actively managed CFM entity associated with a specific DSAP of a service instance, which can generate and receive CFM frames and track any responses. It is an endpoint of a single MA, and terminates a separate maintenance entity for each of the other MEPs in the same MA.

MEP CCDB—A database, maintained by every MEP, that maintains received information about other MEPs in the maintenance domain.

MIP—maintenance intermediate point. A CFM entity, associated with a specific pair of ISS SAPs or EISS Service Access Points, which reacts and responds to CFM frames. It is associated with a single maintenance association and is an intermediate point within one or more maintenance entities.

MIP CCDB—A database of information about the MEPs in the maintenance domain. The MIP CCDB can be maintained by a MIP.

MP—maintenance point. Either a MEP or a MIP.

MPID—maintenance endpoint identifier. A small integer, unique over a given MA, that identifies a specific MEP.

OAM—operations, administration, and maintenance. A term used by several standards bodies to describe protocols and procedures for operating, administrating, and maintaining networks. Examples are ATM OAM and IEEE Std. 802.3ah OAM.

operator—Entity that provides a service provider a single network of provider bridges or a single Layer 2 or Layer 3 backbone network. An operator may be identical to or a part of the same organization as the service provider. For purposes of IEEE P802.1ag, Draft Standard for Local and Metropolitan Area Networks, the operator and service provider are presumed to be separate organizations.

Terms such as “customer,” “service provider,” and “operator” reflect common business relationships among organizations and individuals that use equipment implemented in accordance with IEEE P802.1ag.

UNI—user-network interface. A common term for the connection point between an operator's bridge and customer equipment. A UNI often includes a C-VLAN-aware bridge component. The term UNI is used broadly in the IEEE P802.1ag standard when the purpose for various features of CFM are explained. UNI has no normative meaning.

CHAPTER 4

Configuring Ethernet CFM for the Cisco ASR 1000 Router

IEEE Connectivity Fault Management (CFM) is an end-to-end per-service Ethernet layer Operations, Administration, and Maintenance (OAM) protocol. CFM includes proactive connectivity monitoring, fault verification, and fault isolation for large Ethernet metropolitan-area networks (MANs) and WANs.

This document describes the implementation of IEEE 802.1ag Standard-Compliant CFM (IEEE CFM) and Y.1731 in Cisco IOS XE software for the Cisco ASR 1000 Series Aggregation Services Router. Y.1731 is an ITU-T recommendation for OAM functions in Ethernet-based networks. IEEE CFM and Y.1731 together will be called "Ethernet CFM" throughout this document.

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.

Prerequisites for Configuring Ethernet CFM for the Cisco ASR 1000 Router

- The network topology and network administration have been evaluated.
- Business and service policies have been established.
- EVC associated with CFM domain must be configured with the L2VFI interface command
    - Before configuring CFM over L2VFI ensure EVC and Bridge Domain are configured.
    - Bridge-domain under L2VFI must be configured prior to configuring CFM MEP

Restrictions for Configuring Ethernet CFM for the Cisco ASR 1000 Router

- Ethernet CFM on the Cisco ASR 1000 Series Aggregation Services Router is not compatible with prestandard CFM.
- Locked Signal (ETH-LCK) and Test Signal (ETH-Test) are not supported.
- Link Trace (ETH-LTM/ETH-LTR) over L2VFI is not supported.
- Configuring MIP/MEP under L2VFI is not supported.
- For Connectivity Performance Management functionalities, only single-ended delay (ETH-DM) is supported.
- QinQ encapsulation is not supported on the Cisco ASR 1000 Series Aggregation Services Router for CFM for routed subinterfaces.

Information About Configuring Ethernet CFM for the Cisco ASR 1000 Router

Ethernet CFM

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

Ethernet CFM is distinct from other metro-Ethernet OAM protocols by being an end-to-end technology. For example, Multiprotocol Label Switching (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. Ethernet Local Management Interface (E-LMI) is confined between the user-end provider edge (uPE) and CE and relies on CFM for reporting status of the metro-Ethernet network to the CE.

The benefits of Ethernet CFM are:

- End-to-end service-level OAM technology
- Reduced operating expense for service provider Ethernet networks
- Competitive advantage for service providers

Benefits of Ethernet CFM

- End-to-end service-level OAM technology
- Reduced operating expense for service provider Ethernet networks
- Competitive advantage for service providers

Maintenance Associations

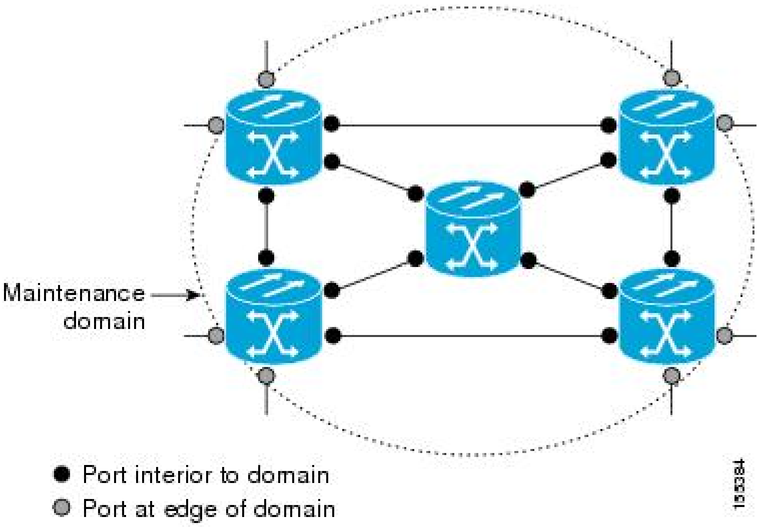
An MA identifies a service that can be uniquely identified within a maintenance domain. There can be many MAs within a domain. The MA direction is specified when the MA is configured. The short MA name must be configured on a domain before MEPs can be configured.

The CFM protocol runs for a specific MA.

Maintenance Domains

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.

Figure 1: 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 among domains. The hierarchical relationship of domains parallels that 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 management approach would be to have the service provider assign maintenance levels to operators.

Ethernet 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 determine domain configurations.

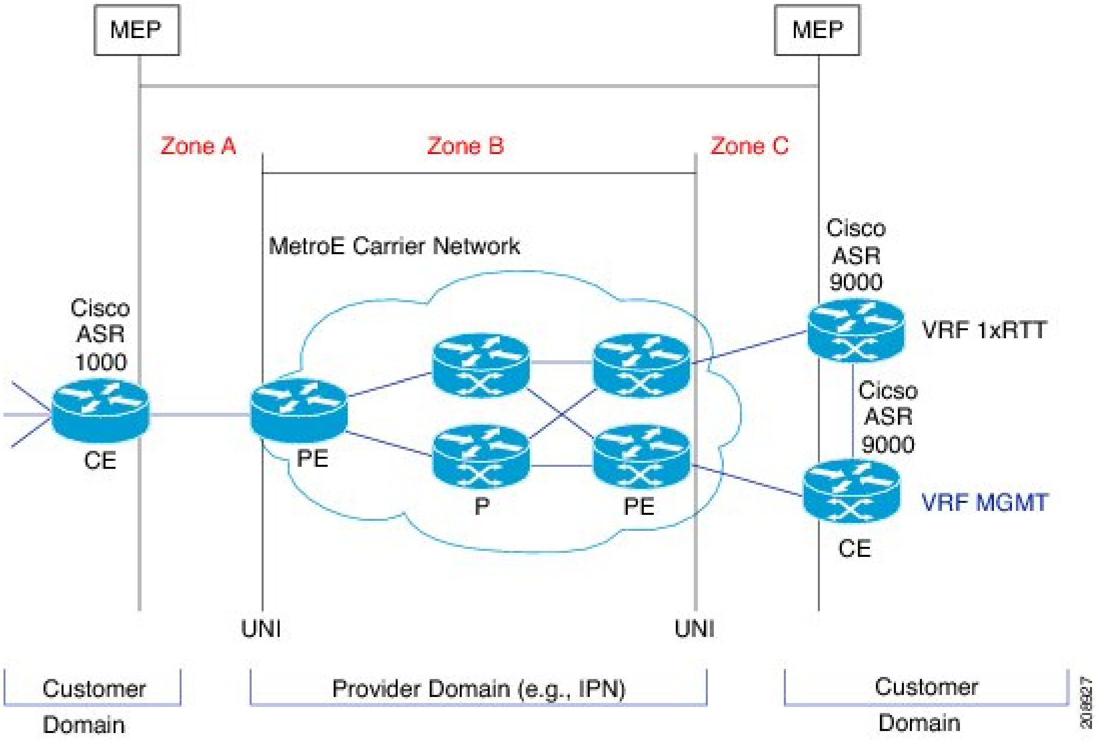
The following characteristics of domains are supported:

- Name is a maximum of 154 characters in length.
- Direction is specified when the MA is configured.
- Down (toward the wire) MEPs.

A domain can be removed when all maintenance points within the domain have been removed and all remote MEP entries in the continuity check database (CCDB) for the domain have been purged.

The figure below illustrates service provider and customer domains and where the Cisco ASR 1000 router is in the network.

Figure 2: Service Provider and Customer Domains



Maintenance Points

A maintenance point (MIP) is a demarcation point on an interface or port that participates in Connectivity Fault Management (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, maintenance end points (MEPs) and MIPs. Support for MIPs varies by Cisco release.

Maintenance Association Endpoints

Maintenance association endpoints (MEPs) reside at the edge of a maintenance domain and confine Ethernet Connectivity Fault Management (CFM) messages within the domain via the maintenance domain level. MEPs periodically transmit and receive continuity check messages (CCMs) from other MEPs within the domain. At the request of an administrator, linktrace and loopback messages can also be transmitted. MEPs are either “Up” (toward the bridge) or “Down” (toward the wire). Support for Up MEPs varies by Cisco release.

When the **continuity-check static rmep** command is configured on a port MEP and continuity checking does not detect a removed MEP, the port is set to MAC operation down and the interface protocol is set to down. Normal traffic is stopped because the line protocol is down, but CFM packets still pass.

MEP configurations can be removed after all pending loopback and traceroute replies are removed and the service on the interface is set to transparent mode.

Down MEPs for Routed Ports

Down MEPs communicate through the wire.

Down MEPs use the port MAC address.

A Down MEP performs the following functions:

- Sends and receives Ethernet CFM frames at its level via the wire connected to the port where the MEP is configured.
- Processes all Ethernet CFM frames at its level coming from the direction of the wire.
- Drops all Ethernet CFM frames at a lower level coming from the direction of the wire.
- Transparently drops all Ethernet CFM frames at a higher level, independent of whether they came in from the bridge or wire.

Ethernet CFM Messages

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

Bridges that cannot interpret Ethernet CFM messages forward them as normal data frames. All Ethernet CFM messages are confined to a maintenance domain and to an MA. Three types of messages are supported:

- Continuity Check
- Linktrace
- Loopback

Continuity Check Messages

Ethernet CFM continuity check messages (CCMs) are multicast heartbeat messages exchanged periodically among MEPs. They allow MEPs to discover other MEPs within a domain. CCMs are confined to a domain.

CFM CCMs have the following characteristics:

- Transmitted at a periodic interval by MEPs. The minimum interval is milliseconds (ms).
- Terminated by remote MEPs at the same maintenance level.
- Unidirectional and do not solicit a response.
- Indicate the status of the interface on which the MEP is configured.

Linktrace Messages

Ethernet CFM linktrace messages (LTMs) 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 are similar to Layer 3 traceroute messages. LTMs allow the transmitting node to discover vital connectivity data about the path. LTMs are intercepted by maintenance points along the path and processed, transmitted, or dropped. At each hop where there is a maintenance point at the same level, a linktrace message reply (LTR) is transmitted back to the originating MEP. For each visible MIP, linktrace messages indicate ingress action, relay action, and egress action.

Linktrace 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. LTMs are multicast and LTRs are unicast.

Loopback Messages

Ethernet CFM loopback messages (LBMs) 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 (LBR) 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.

Because LBMs are unicast, they are forwarded like normal data frames except with the maintenance level restriction. If the outgoing port is known in the bridge’s forwarding database and allows Ethernet CFM frames at the message’s maintenance level to pass through, the frame is sent out on that port. If the outgoing port is unknown, the message is broadcast on all ports in that domain.

An Ethernet CFM LBM can be generated on demand using the CLI. The source of a loopback message must be a MEP. Both Ethernet CFM LBMs and LBRs are unicast, and LBMs specify the destination MAC address or MEP identifier (MPID), VLAN, and maintenance domain.

Cross-Check Function

The cross-check function is a timer-driven postprovisioning service verification between dynamically discovered MEPs (via continuity check messages CCMs)) and expected MEPs (via configuration) for a service. The cross-check function verifies that all endpoints of a multipoint or point-to-point service are operational. The function supports notifications when the service is operational; otherwise it provides alarms and notifications for unexpected or missing endpoints.

The cross-check function is performed one time. You must initiate the cross-check function from the CLI every time you want a service verification.

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
```

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)#
```

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
```

HA Feature Support in Ethernet CFM

In access and service provider networks using Ethernet technology, High availability (HA) is a requirement. End-to-end connectivity status information is critical and must be maintained on a hot standby Route Processor (RP).

Note A hot standby RP 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 CE, PE, and access aggregation PE (uPE) network nodes based on information received by protocols such as Ethernet local management interface (LMI) and CFM, and 802.3ah. This status information is used to either stop traffic or switch to backup paths when an interface is down.

Every transaction involves either accessing or updating data among various databases. If the database is synchronized across active and standby modules, the modules are transparent to clients.

The Cisco infrastructure provides various component application program interfaces (APIs) that help to maintain a hot standby RP. Metro Ethernet HA clients CFM HA and in-service software upgrades (ISSU) interact with these components, update the database, and trigger necessary events to other components.

Benefits of CFM HA

- Elimination of network downtime for Cisco software image upgrades, allowing for faster upgrades that result in high availability.
- Elimination of resource scheduling challenges associated with planned outages and late night maintenance windows.
- Accelerated deployment of new services and applications and facilitation of faster implementation of new features, hardware, and fixes than if HA was not supported.
- Reduced operating costs due to outages while delivering high service levels.

- CFM updates its databases and controls its own HA messaging and versioning, and this control facilitates maintenance.

NSF SSO Support in Ethernet CFM

The redundancy configurations SSO and NSF are both supported in Ethernet CFM and are automatically enabled. A switchover from an active to a standby RP occurs when the active RP fails, is removed from the networking device, or is manually taken down for maintenance. NSF interoperates with the SSO feature to minimize network downtime following a switchover. The primary function of Cisco NSF is to continue forwarding packets following an RP switchover.

For detailed information about SSO, see the “Stateful Switchover” module of the *Cisco IOS High Availability Configuration Guide*. For detailed information about the NSF feature, see the “Cisco Nonstop Forwarding” module of the *High Availability Configuration Guide*.

ISSU Support in Ethernet CFM

In Service Upgrades (ISSU) allows you to perform a Cisco software upgrade or downgrade without disrupting packet flow. Ethernet Connectivity Fault Management (CFM) performs a bulk update and a runtime update of the continuity check database to the standby route processor (RP), including adding, deleting, or updating a row. This checkpoint data requires ISSU capability to transform messages from one release to another. All the components that perform active RP to standby RP updates using messages require ISSU support.

ISSU is automatically enabled in Ethernet CFM and 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 “Cisco IOS In Service Software Upgrade Process” module of the *High Availability Configuration Guide*.

How to Configure Ethernet CFM for the Cisco ASR 1000 Router

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.

SUMMARY STEPS

1. Determine operator level MIPs.
2. Determine operator level MEPs.
3. Determine service provider MIPs.
4. Determine service provider MEPs.
5. Determine customer MIPs.
6. Determine customer MEPs.

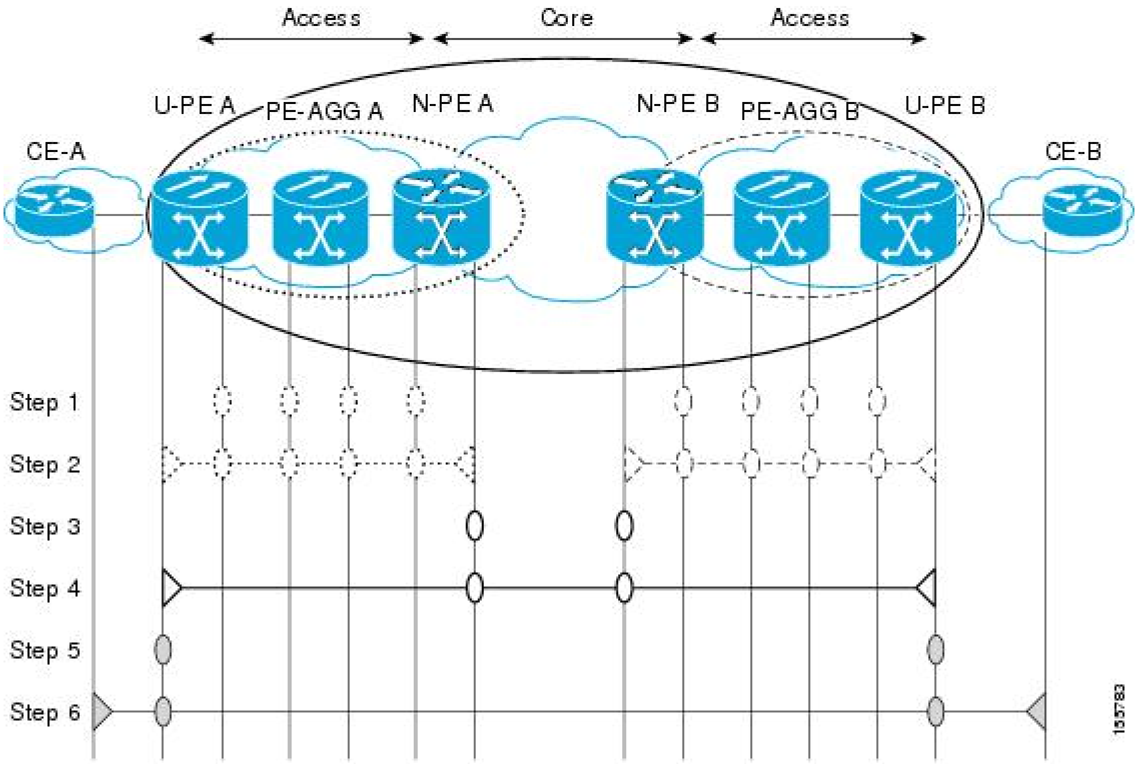
DETAILED STEPS

	Command or Action	Purpose
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).

	Command or Action	Purpose
		• 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.

Examples

The figure below shows an example of a network with a service provider and two operators, A and B. Three domains are to be established to map to each operator and the service provider. In this example, for simplicity we assume that the network uses Ethernet transport end to end. CFM, however, can be used with other transports.



Configuring Ethernet CFM

Provisioning the Network (CE-A)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **mep archive-hold-time** *minutes*
5. **exit**
6. **ethernet cfm global**
7. **ethernet cfm ieee**
8. **ethernet cfm traceroute cache**
9. **ethernet cfm traceroute cache size** *entries*
10. **ethernet cfm traceroute cache hold-time** *minutes*
11. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
12. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown**] [**mep-missing**] [**service-up**]
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `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	**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 5	**exit** **Example:** `Device(config-ecfm)# exit`	Returns the device to global configuration mode.
Step 6	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 7	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the CFM IEEE version of CFM. • This command is automatically issued when the **ethernet cfm global** command is issued.
Step 8	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 9	**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.

	Command or Action	Purpose
Step 10	**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 11	**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 12	**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`	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 13	**end**
	Example:
	`Device(config)# end`
		Returns the device to privileged EXEC mode.

Provisioning the Network (CE-B)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **mep archive-hold-time** *minutes*
5. **exit**
6. **ethernet cfm global**
7. **ethernet cfm ieee**
8. **ethernet cfm traceroute cache**
9. **ethernet cfm traceroute cache size** *entries*
10. **ethernet cfm traceroute cache hold-time** *minutes*
11. **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]
12. **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown**] [**mep-missing**] [**service-up**]
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `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 an outward CFM maintenance domain at a specified level and enters Ethernet CFM configuration mode.
Step 4	**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 5	**exit** **Example:** `Device(config-ecfm)# exit` **Example:**	Returns the device to global configuration mode.
Step 6	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 7	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the CFM IEEE version of CFM. • This command is automatically issued when the **ethernet cfm global** command is issued.
Step 8	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 9	**ethernet cfm traceroute cache size** *entries* **Example:**	Sets the maximum size for the CFM traceroute cache table.

	Command or Action	Purpose
	`Device(config)# ethernet cfm traceroute cache size 200`	
Step 10	**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 11	**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 mep-up, mep-down, config, loop, and cross-connect events.
Step 12	**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`	Enables SNMP trap generation for Ethernet CFM mep-unknown, mep-missing, and service-up continuity check events in relation to the cross-check operation between statically configured MEPs and those learned via CCMs.

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

Provisioning Service (CE-A)

Perform this task to set up service for Ethernet CFM. Optionally, when this task is completed, you may configure and enable the cross-check function. To perform this optional task, see "Configuring and Enabling the Cross-Check Function (CE-A)".

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]]
5. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
6. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
7. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
8. **exit**
9. **mep archive-hold-time** *minutes*

10. **exit**
11. **ethernet cfm global**
12. **ethernet cfm ieee**
13. **ethernet cfm traceroute cache**
14. **ethernet cfm traceroute cache size** *entries*
15. **ethernet cfm traceroute cache hold-time** *minutes*
16. **interface** *type number*
17. **ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**port** | **vlan** *vlan-id*}
18. **ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**port** | **vlan** *vlan-id*}
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Router(config)# ethernet cfm domain Customer level 7`	Defines a CFM maintenance domain at a specified maintenance level and enters Ethernet CFM configuration mode.
Step 4	**service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]] **Example:** `Router(config-ecfm)# service Customer1 vlan 101 direction down`	Configures an MA within a maintenance domain and enters CFM service configuration mode. • If a service is already configured and you configure a new MA name and also specify the **direction down** keyword, a second service is added that maps to the same VLAN. If you configure a new MA name and do not specify the **direction down** keyword, the service is renamed to the new MA name.
Step 5	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:** `Router(config-ecfm-srv)# continuity-check`	Enables the transmission of CCMs.
Step 6	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]	Configures the time period between CCM transmissions. • The values supported are platform dependent.

	Command or Action	Purpose
	Example: `Router(config-ecfm-srv)# continuity-check interval 10s`	
Step 7	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:** `Router(config-ecfm-srv)# continuity-check loss-threshold 10`	Sets the number of CCMs that should be missed before declaring that a remote MEP is down.
Step 8	**exit** **Example:** `Router(config-ecfm-srv)# exit` **Example:**	Returns the device to Ethernet CFM configuration mode.
Step 9	**mep archive-hold-time** *minutes* **Example:** `Router(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 10	**exit** **Example:** `Router(config-ecfm)# exit`	Returns the device to global configuration mode.
Step 11	**ethernet cfm global** **Example:** `Router(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 12	**ethernet cfm ieee** **Example:** `Router(config)# ethernet cfm ieee`	Enables the CFM IEEE version of CFM. • This command is automatically issued when the **ethernet cfm global** command is issued.
Step 13	**ethernet cfm traceroute cache** **Example:** `Router(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 14	**ethernet cfm traceroute cache size** *entries* **Example:** `Router(config)# ethernet cfm traceroute cache size 200`	Sets the maximum size for the CFM traceroute cache table.

	Command or Action	Purpose
Step 15	**ethernet cfm traceroute cache hold-time** *minutes* **Example:** `Router(config)# ethernet cfm traceroute cache hold-time 60`	Sets the amount of time that CFM traceroute cache entries are retained.
Step 16	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/3`	Specifies an interface and enters interface configuration mode.
Step 17	**ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**port** | **vlan** *vlan-id*} **Example:** `Router(config-if)# ethernet cfm mep domain Customer mpid 701 vlan 100`	Sets a port as internal to a maintenance domain and defines it as a MEP.
Step 18	**ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**port** | **vlan** *vlan-id*} **Example:** `Router(config-if)# ethernet cfm mep domain Customer mpid 701 vlan 100`	Sets a port as internal to a maintenance domain and defines it as a MEP.

Step 19	**end**
	Example:
	`Router(config-if)# end`
		Returns the device to privileged EXEC mode.

Provisioning Service (CE-B)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **mep archive-hold-time** *minutes*
5. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
6. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
7. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
8. **exit**
9. **exit**
10. **ethernet cfm global**
11. **ethernet cfm ieee**
12. **ethernet cfm traceroute cache**

13. **ethernet cfm traceroute cache size** *entries*
14. **ethernet cfm traceroute cache hold-time** *minutes*
15. **interface** *slot*/*subslot*/*port*
16. **ethernet cfm mep level** *level-id* [**inward** | **outward domain** *domain-name*] **mpid** *id* **vlan** {**any** | *vlan-id* | **,** *vlan-id* | *vlan-id* **-** *vlan-id* | **,** *vlan-id* **-** *vlan-id*}
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain Customer level 7`	Defines a CFM maintenance domain at a specified level and enters Ethernet CFM configuration mode.
Step 4	**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 5	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:** `Device(config-ecfm-srv)# continuity-check`	Enables the transmission of CCMs.
Step 6	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:** `Device(config-ecfm-srv)# continuity-check interval 10s`	Configures the time period between CCM transmissions. • The values supported are platform dependent.
Step 7	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:**	Sets the number of CCMs that should be missed before declaring that a remote MEP is down.

	Command or Action	Purpose
	`Device(config-ecfm-srv)# continuity-check loss-threshold 10`	
Step 8	**exit** **Example:** `Device(config-ecfm-srv)# exit`	Returns the device to Ethernet CFM configuration mode.
Step 9	**exit** **Example:** `Device(config-ecfm)# exit`	Returns the device to global configuration mode.
Step 10	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 11	**ethernet cfm ieee** **Example:** `Device(config)# ethernet cfm ieee`	Enables the 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	**interface** *slot*/*subslot*/*port* **Example:**	Specifies an interface and enters interface configuration mode.
Step 16	**ethernet cfm mep level** *level-id* [**inward** | **outward domain** *domain-name*] **mpid** *id* **vlan** {**any** | *vlan-id* | **,** *vlan-id* | *vlan-id* **-** *vlan-id* | **,** *vlan-id* **-** *vlan-id*} **Example:**	Provisions an interface as a domain boundary.

	Command or Action	Purpose
	`Device(config-if)# ethernet cfm mep level 7 outward domain Customer mpid 701 vlan 100`	

Step 17	**end**
	Example:
	`Device(config-if)# end`
	Example:
	`Device#`
		Returns the device to privileged EXEC mode.

Configuring and Enabling the Cross-Check Function (CE-A)

Perform this task to configure and enable cross-checking for a down MEP. This task requires you to configure and enable cross-checking on two devices. This task is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **mep mpid** *mpid*
6. **exit**
7. **ethernet cfm mep crosscheck start-delay** *delay*
8. **exit**
9. **ethernet cfm mep crosscheck** {**enable** | **disable**} **domain** *domain-name* {**port** | **vlan** {*vlan-id* | *vlan-id* - *vlan-id* | **,** *vlan-id* - *vlan-id*}}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:**	Defines a CFM domain at a specified level and enters Ethernet CFM configuration mode.

	Command or Action	Purpose
	`Device(config)# ethernet cfm domain Customer level 7`	
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.
Step 5	**mep mpid** *mpid* **Example:** `Device(config-ecfm)# mep mpid 702`	Statically defines the MEPs within a maintenance association.
Step 6	**exit** **Example:** `Device(config-ecfm)# exit`	Returns the device to global configuration mode.
Step 7	**ethernet cfm mep crosscheck start-delay** *delay* **Example:** `Device(config)# ethernet cfm mep crosscheck start-delay 60`	Configures the maximum amount of time that the device waits for remote MEPs to come up before the cross-check operation is started.
Step 8	**exit** **Example:** `Device(config)# exit`	Returns the device to privileged EXEC mode.
Step 9	**ethernet cfm mep crosscheck** {**enable** | **disable**} **domain** *domain-name* {**port** | **vlan** {*vlan-id* | *vlan-id* **-** *vlan-id* | **,** *vlan-id* **-** *vlan-id*}} **Example:** `Device# ethernet cfm mep crosscheck enable domain cust4 vlan 100`	Enables cross-checking between the list of configured remote MEPs of a domain and MEPs learned through CCMs.

Configuring and Enabling the Cross-Check Function (CE-B)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**
5. **mep mpid** *mpid*

6. **exit**
7. **ethernet cfm mep crosscheck start-delay** *delay*
8. **exit**
9. **ethernet cfm mep crosscheck** {**enable** | **disable**} **domain** *domain-name* {**port** | **vlan** {*vlan-id* | *vlan-id* - *vlan-id* | **,** *vlan-id* - *vlan-id*}}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain Customer level 7`	Defines an outward CFM domain at a specified 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.
Step 5	**mep mpid** *mpid* **Example:** `Device(config-ecfm)# mep mpid 702`	Statically defines the MEPs within a maintenance association.
Step 6	**exit** **Example:** `Device(config-ecfm)# exit`	Returns the device to global configuration mode.
Step 7	**ethernet cfm mep crosscheck start-delay** *delay* **Example:** `Device(config)# ethernet cfm mep crosscheck start-delay 60`	Configures the maximum amount of time that the device waits for remote MEPs to come up before the cross-check operation is started.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config)# exit`	Returns the device to privileged EXEC mode.
Step 9	**ethernet cfm mep crosscheck** {**enable** | **disable**} **domain** *domain-name* {**port** | **vlan** {*vlan-id* | *vlan-id* **-** *vlan-id* | **,** *vlan-id* **-** *vlan-id*}} **Example:** `Device# ethernet cfm mep crosscheck enable domain cust4 vlan 100`	Enables cross-checking between the list of configured remote MEPs of a domain and MEPs learned through CCMs.

Configuration Examples for Configuring Ethernet CFM for the Cisco ASR 1000 Router

The following two examples show configurations for a network. Configurations are shown not only for the Carrier Ethernet Cisco ASR 1000 Series Aggregation Services Routers, but also for the devices used at the access and core of the service provider’s network.

Example: Provisioning a Network

This configuration example shows only CFM-related commands. All commands that are required to set up the data path and configure the VLANs on the device are not shown. However, it should be noted that CFM traffic will not flow into or out of the device if the VLANs are not properly configured.

CE-A Configuration

```
!
ethernet cfm global
ethernet cfm ieee
!
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm mip auto-create level 7 vlan 1-4094
!
interface
 ethernet cfm mip level 7 vlan 101   <<<< Manual MIP
 ethernet cfm mep domain ServiceProvider-L4 mpid 401 vlan 101
 ethernet cfm mep domain OperatorA-L1 mpid 101 vlan 101
!
interface
 ethernet cfm mip level 1 vlan 101   <<<< Manual MIP
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

U-PE A Configuration

```
!
ethernet cfm global
ethernet cfm ieee
!
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm mip auto-create level 7 vlan 1-4094
!
interface
 ethernet cfm mip level 7 vlan 101   <<<< Manual MIP
 ethernet cfm mep domain ServiceProvider-L4 mpid 401 vlan 101
 ethernet cfm mep domain OperatorA-L1 mpid 101 vlan 101
!
interface
 ethernet cfm mip level 1 vlan 101   <<<< Manual MIP
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

PE-AGG A Configuration

```
ethernet cfm global
ethernet cfm ieee
ethernet cfm domain OperatorA-L1 level 1
mep archive-hold-time 65
  mip auto-create
  service MetroCustomer1OpA vlan 101
!
interface
 ethernet cfm mip level 1 vlan 101   <<<< Manual MIP
!
interface
 ethernet cfm mip level 1    <<<< Manual MIP
```

N-PE A Configuration

```
!
ethernet cfm global
ethernet cfm ieee
!
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain ServiceProvider-L4 level 4
 mep archive-hold-time 60
 mip auto-create
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
 mip auto-create
service MetroCustomer1OpA vlan 101
  continuity-check
!
interface
 ethernet cfm mip level 1    <<<< manual MIP
!
interface
```

```
 ethernet cfm mip level 4    <<<< manual MIP
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

U-PE B Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain Customer-L7 level 7
 mip auto-create
 service Customer1 vlan 101 direction down
!
ethernet cfm domain ServiceProvider-L4 level 4
 mep archive-hold-time 60
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorB level 2
 mip auto-create
 mep archive-hold-time 65
 service MetroCustomer1OpB vlan 101
  continuity-check
!
interface
 ethernet cfm mip level 7   <<<< manual MIP
!
interface
 ethernet cfm mip level 2   <<<< manual MIP
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

PE-AGG B Configuration

```
ethernet cfm global
ethernet cfm ieee
!
ethernet cfm domain OperatorB level 2
 mep archive-hold-time 65
 mip auto-create
 service MetroCustomer1OpB vlan 101
!
interface
 ethernet cfm mip level 2   <<<< manual MIP
!
interface
 ethernet cfm mip level 2   <<<< manual MIP
```

N-PE B Configuration

```
!
ethernet cfm global
ethernet cfm ieee
!
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
```

```
!
ethernet cfm domain ServiceProvider level 4
 mep archive-hold-time 60
 mip auto-create
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorB level 2
 mep archive-hold-time 65
 mip auto-create
 service MetroCustomer1OpB vlan 101
  continuity-check
!
interface
ethernet cfm mip level 2    <<<< manual MIP
!
interface
 ethernet cfm mip level 4   <<<< manual MIP
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

CE-B Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain Customer-L7 level 7
 service Customer1 vlan 101 direction down
  continuity-check
!
snmp-server enable traps ethernet cfm cc mep-up mep-down cross-connect loop config
snmp-server enable traps ethernet cfm crosscheck mep-missing mep-unknown service-up
```

Example: Provisioning Service

CE-A Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain Customer-L7 level 7
 service Customer1 vlan 101 direction down
  continuity-check
!
interface
 ethernet cfm mep domain Customer-L7 mpid 701 vlan 101
```

U-PE A Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm mip auto-create level 7 vlan 1-4094
!
ethernet cfm domain ServiceProvider-L4 level 4
 mep archive-hold-time 60
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorA-L1 level 1
 mep archive-hold-time 65
 mip auto-create
 service MetroCustomer1OpA vlan 101
  continuity-check
!
interface
 ethernet cfm mip level 7 vlan 101   <<<< Manual MIP
 ethernet cfm mep domain ServiceProvider-L4 mpid 401 vlan 101
 ethernet cfm mep domain OperatorA-L1 mpid 101 vlan 101
!
interface
 ethernet cfm mip level 1 vlan 101   <<<< Manual MIP
```

PE-AGG A Configuration

```
ethernet cfm global
ethernet cfm ieee
ethernet cfm domain OperatorA-L1 level 1
mep archive-hold-time 65
  mip auto-create
  service MetroCustomer1OpA vlan 101
!
interface
 ethernet cfm mip level 1 vlan 101   <<<< Manual MIP
!
interface
 ethernet cfm mip level 1    <<<< Manual MIP
```

N-PE A Configuration

```
!
ethernet cfm global
ethernet cfm ieee
!
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain ServiceProvider-L4 level 4
 mep archive-hold-time 60
 mip auto-create
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
 mip auto-create
service MetroCustomer1OpA vlan 101
  continuity-check
!
interface
```

```
 ethernet cfm mip level 1    <<<< manual MIP
!
interface
 ethernet cfm mip level 4    <<<< manual MIP
 ethernet cfm mep domain OperatorA mpid 102 vlan 101
```

U-PE B Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain Customer-L7 level 7
 mip auto-create
 service Customer1 vlan 101 direction down
!
ethernet cfm domain ServiceProvider-L4 level 4
 mep archive-hold-time 60
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorB level 2
 mep archive-hold-time 65
 service MetroCustomer1OpB vlan 101
  continuity-check
!
interface
 ethernet cfm mip level 7   <<<< manual MIP
 ethernet cfm mep domain ServiceProvider-L4 mpid 402 vlan 101
 ethernet cfm mep domain OperatorB mpid 201 vlan 101
!
interface
 ethernet cfm mip level 2   <<<< manual MIP
```

N-PE B Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain ServiceProvider level 4
 mep archive-hold-time 60
 mip auto-create
 service MetroCustomer1 vlan 101
  continuity-check
!
ethernet cfm domain OperatorB level 2
 mep archive-hold-time 65
 mip auto-create
 service MetroCustomer1OpB vlan 101
  continuity-check
!
interface
ethernet cfm mip level 2      <<<< manual MIP
!
interface
```

```
 ethernet cfm mip level 4     <<<< manual MIP
 ethernet cfm mep domain OperatorB mpid 202 vlan 101
```

CE-B Configuration

```
!
ethernet cfm global
ethernet cfm ieee
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
ethernet cfm domain Customer-L7 level 7
 service Customer1 vlan 101 direction down
  continuity-check
!
interface
 ethernet cfm mep domain Customer-L7 mpid 702 vlan 101
```

Additional References

Related Documents

Related Topic	Document Title
CFM commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*
Configuring IEEE Standard-Compliant Ethernet CFM in a Service Provider Network	"Configuring IEEE Standard-Compliant Ethernet CFM in a Service Provider Network"
IP SLAs for Metro Ethernet	"IP SLAs for Metro Ethernet"
ISSU feature and functions	"Cisco IOS Broadband High Availability In Service Software Upgrade"
Performing an ISSU	"Cisco IOS In Service Software Upgrade Process and Enhanced Fast Software Upgrade Process"
SSO	"Stateful Switchover" module of the *High Availability Configuration Guide*

Standards

Standard	Title
IEEE 802.1ag Standard	*802.1ag - 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	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Technical Assistance

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

Feature Information for Configuring Ethernet CFM for the Cisco ASR 1000 Router

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

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

Table 2: Feature Information for Configuring Ethernet CFM for the Cisco ASR 1000 Router

Feature Name	Releases	Feature Information
IEEE 802.1ag-2007 Compliant CFM for ASR1000	Cisco IOS XE Release 3.2S	IEEE CFM is an end-to-end per-service Ethernet layer OAM protocol. CFM includes proactive connectivity monitoring, fault verification, and fault isolation for large Ethernet metropolitan-area networks (MANs) and WANs. Y.1731 is an ITU-T recommendation for OAM functions in Ethernet-based networks. This feature is the implementation of IEEE 802.1ag Standard-Compliant CFM and Y.1731 in Cisco IOS XE software. The following commands were introduced or modified: **continuity-check**, **ethernet cfm domain level**, **ethernet cfm global**, **ethernet cfm ieee**, **ethernet cfm mep crosscheck**, **ethernet cfm mep crosscheck start-delay**, **ethernet cfm mep domain mpid**, **ethernet cfm traceroute cache**, **ethernet cfm traceroute cache hold-time**, **ethernet cfm traceroute cache size**, **mep archive-hold-time**, **mep mpid**, **service (cfm-srv)**, **snmp-server enable traps ethernet cfm cc**, and **snmp-server enable traps ethernet cfm crosscheck**.
E-OAM : Multiple port MAs under single MD	Cisco IOS XE Release 3.7S	Support for multiple MAs under a single maintenance domain was added. The following commands were introduced or modified: **clear ethernet cfm ais**, **ethernet cfm lck**, **ethernet cfm mep crosscheck**, **ethernet cfm mep domain mpid**, **ping ethernet**, **show ethernet cfm maintenance-points remote**, **show ethernet cfm maintenance-points remote crosscheck**, **show ethernet cfm maintenance-points remote detail**, **show ethernet cfm traceroute-cache**, **traceroute ethernet**.

Glossary

CCM—continuity check message. A multicast CFM frame that a MEP transmits periodically to ensure continuity across the maintenance entities to which the transmitting MEP belongs, at the MA level on which the CCM is sent. No reply is sent in response to receiving a CCM.

EVC—Ethernet virtual connection. An association of two or more user-network interfaces.

fault alarm—An out-of-band signal, typically an SNMP notification, that notifies a system administrator of a connectivity failure.

inward-facing MEP—A MEP that resides in a bridge and transmits to and receives CFM messages from the direction of the bridge relay entity.

maintenance domain—The network or part of the network belonging to a single administration for which faults in connectivity are to be managed. The boundary of a maintenance domain is defined by a set of DSAPs, each of which may become a point of connectivity to a service instance.

maintenance domain name—The unique identifier of a domain that CFM is to protect against accidental concatenation of service instances.

MEP—maintenance endpoint. An actively managed CFM entity associated with a specific DSAP of a service instance, which can generate and receive CFM frames and track any responses. It is an endpoint of a single MA, and terminates a separate maintenance entity for each of the other MEPs in the same MA.

MEP CCDB—A database, maintained by every MEP, that maintains received information about other MEPs in the maintenance domain.

MIP—maintenance intermediate point. A CFM entity, associated with a specific pair of ISS SAPs or EISS Service Access Points, which reacts and responds to CFM frames. It is associated with a single maintenance association and is an intermediate point within one or more maintenance entities.

MIP CCDB—A database of information about the MEPs in the maintenance domain. The MIP CCDB can be maintained by a MIP.

MP—maintenance point. Either a MEP or a MIP.

MPID—maintenance endpoint identifier. A small integer, unique over a given MA, that identifies a specific MEP.

OAM—operations, administration, and maintenance. A term used by several standards bodies to describe protocols and procedures for operating, administrating, and maintaining networks. Examples are ATM OAM and IEEE Std. 802.3ah OAM.

operator—Entity that provides a service provider a single network of provider bridges or a single Layer 2 or Layer 3 backbone network. An operator may be identical to or a part of the same organization as the service provider. For purposes of IEEE P802.1ag, Draft Standard for Local and Metropolitan Area Networks, the operator and service provider are presumed to be separate organizations.

Terms such as “customer,” “service provider,” and “operator” reflect common business relationships among organizations and individuals that use equipment implemented in accordance with IEEE P802.1ag.

UNI—user-network interface. A common term for the connection point between an operator's bridge and customer equipment. A UNI often includes a C-VLAN-aware bridge component. The term UNI is used broadly in the IEEE P802.1ag standard when the purpose for various features of CFM are explained. UNI has no normative meaning.

CHAPTER 5

Configuring Ethernet Virtual Connections on the Cisco ASR 1000 Series Router

Ethernet virtual circuit (EVC) infrastructure is a Layer 2 platform-independent bridging architecture that supports Ethernet services. This document describes the infrastructure and the features it supports on the Cisco ASR 1000 Series Aggregation Services Router.

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 Configuring EVCs on the Cisco ASR 1000 Series Router

- Bridge domain configuration is supported only as part of the EVC service instance configuration.
- The following features are not supported:
  - Service instance (Ethernet flow point [EFP]) group support
  - EVC cross-connect and connect forwarding services

- Ethernet service protection (Ethernet Operations, Administration, and Maintenance [EOAM], connectivity fault management [CFM], Ethernet Local Management Interface [E-LMI]) on EVCs
- IPv6 access control lists (ACLs) are not supported.

Information About Configuring EVCs on the Cisco ASR 1000 Series Router

The following topics are described in this section and provide background information for configuring EVCs on the Cisco ASR 1000 Series Router:

In Cisco IOS XE Release 3.2S and later releases, the following features are supported in the EVC infrastructure:

In Cisco IOS XE Release 3.3S, Layer 3 and Layer 4 protocol support was added. This support is described in the "Layer 3 and Layer 4 ACL Support".

EVCs

An 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.

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 switch port VLAN translation model.

Service Instances and Associated EFPs

Configuring a service instance on a Layer 2 port creates a pseudoport or EFP on which you configure EVC features. Each service instance has a unique number per interface, but you can use the same number on different interfaces because service instances on different ports are not related.

An EFP classifies frames from the same physical port to one of the multiple service instances associated with that port, based on user-defined criteria. Each EFP can be associated with different forwarding actions and behavior.

When an EFP is created, the initial state is UP. The state changes to DOWN under the following circumstances:

- The EFP is explicitly shut down by a user.
- The main interface to which the EFP is associated is down or removed.
- If the EFP belongs to a bridge domain, the bridge domain is down.
- The EFP is forced down as an error-prevention measure of certain features.

Use the **service instance ethernet** interface configuration command to create an EFP on a Layer 2 interface and to enter service instance configuration mode. Service instance configuration mode is used to configure all management and control data plane attributes and parameters that apply to the service instance on a per-interface basis. The service instance number is the EFP identifier.

After the device enters service instance configuration mode, you can configure these options:

- default--Sets a command to its defaults
- description--Adds a service instance-specific description
- encapsulation--Configures Ethernet frame match criteria
- exit--Exits from service instance configuration mode
- no--Negates a command or sets its defaults
- shutdown--Takes the service instance out of service

Encapsulation (Flexible Service Mapping)

Encapsulation defines the matching criteria that map a VLAN, a range of VLANs, class of service (CoS) bits, Ethertype, or a combination of these to a service instance. VLAN tags and CoS can be a single value, a range, or a list. Ethertype can be a single type or a list of types.

Different types of encapsulations are default, dot1ad, dot1q, priority-tagged, and untagged. On the Cisco ASR 1000 Series Router, priority-tagged frames are always single-tagged. Valid Ethertypes (type) are ipv4, ipv6, pppoe-all, pppoe-discovery, and pppoe-session.

Encapsulation classification options also include:

- inner tag CoS
- inner tag VLAN
- outer tag CoS
- outer tag VLAN
- outer tag Ethertype (VLAN type)--VLAN type is always matched. If you do not specify an alternative, the default is 0x8100 for dot1q and 0x88a8 for dot1ad.
- payload Ethertype--Any Ethertype tag after the VLAN tag

When you configure an encapsulation method, you enable flexible service mapping, which allows you to map an incoming packet to an EFP based on the configured encapsulation.

The default behavior for flexible service mapping based on outer 802.1q and 802.1ad VLAN tag values is nonexact, meaning that when the EFP encapsulation configuration does not explicitly specify an inner (second) VLAN tag matching criterion, the software maps both single-tagged and double-tagged frames to the EFP as long as the frames fulfill the criteria of outer VLAN tag values. The command-line interface (CLI) does allow you to specify exact mapping with the **exact** keyword. If this keyword is specified, the EFP is designated as single-tagged-frame-only and double-tagged frames are not classified to that EFP.

Using the CLI **encapsulation** command in service-instance configuration mode, you can set encapsulation criteria. You must configure one encapsulation command per EFP (service instance). After you have configured an encapsulation method, these commands are available in service instance configuration mode:

- **bridge-domain** --Configures a bridge domain.
- **rewrite** --Configures Ethernet rewrite criteria.

The table below shows the supported encapsulation types.

Table 3: Supported Encapsulation Types

Command	Description
encapsulation dot1q *vlan-id* [**,** *vlan-id* [**-** *vlan-id*]]	Defines the matching criteria to be used to map 802.1q frames ingressing on an interface to the appropriate EFP. The options are a single VLAN, a range of VLANs, or lists of VLANs or VLAN ranges. VLAN IDs are 1 to 4094. • Enter a single VLAN ID for an exact match of the outermost tag. • Enter a VLAN range for a ranged outermost match.
encapsulation dot1q *vlan-id* **second-dot1q** *vlan-id* [**,** *vlan-id* [**-** *vlan-id*]]	Double-tagged 802.1q encapsulation. Matching criteria to be used to map QinQ frames ingressing on an interface to the appropriate EFP. The outer tag is unique and the inner tag can be a single VLAN, a range of VLANs or lists of VLANs or VLAN ranges. • Enter a single VLAN ID in each instance for an exact match of the outermost two tags. • Enter a VLAN range for second-dot1q for an exact outermost tag and a range for a second tag.
encapsulation dot1q {**any** | *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]]} **etype** *ethertype*	Ethertype encapsulation is the payload encapsulation type after VLAN encapsulation. Ethertype encapsulation matches any or an exact outermost VLAN or VLAN range and a payload ethertype. Valid values for *ethertype* are **ipv4**, **ipv6**, **pppoe-discovery**, **pppoe-session**, or **pppoe-all**.
encapsulation dot1q *vlan-id* **cos** *cos-value* **second-dot1q** *vlan-id* **cos** *cos-value*	CoS value encapsulation defines match criteria after including the CoS for the S-Tag and the C-Tag. The CoS value is a single digit between 1 and 7 for S-Tag and C-Tag. You cannot configure CoS encapsulation with the **encapsulation untagged** command, but you can configure it with the **encapsulation priority-tagged** command. The result is an exact outermost VLAN and CoS match and second tag. You can also use VLAN ranges.
encapsulation dot1q any	Matches any packet with one or more VLANs.
encapsulation dot1q vlan-type	Specifies the value of the VLAN protocol type, which is the tag protocol identifier (TPID) of an 802.1q VLAN tag. If there is more than one tag, this command refers to the outermost tag. By default the TPID is assumed to be 0x8100. Use this command to set the TPID to other supported alternatives: 0x88A8, 0x9100, 0x9200.

Command	Description
encapsulation dot1ad	Defines the matching criteria to be used to map 802.1d frames ingressing on an interface to the appropriate EFP.
encapsulation untagged	Matching criteria to be used to map native Ethernet frames (without a dot1q tag) entering an interface to the appropriate EFP. Only one EFP per port can have untagged encapsulation. However, a port that hosts EFP matching untagged traffic can also host other EFPs that match tagged frames.
encapsulation default	Configures the default EFP on an interface, acting as a catch-all encapsulation for all packets without a configured encapsulation. All packets are seen as native. If you enter the **rewrite** command with encapsulation default, the command is rejected. Only one default EFP per interface can be configured. If you try to configure more than one default EFP, the command is rejected.
encapsulation priority-tagged	Specifies priority-tagged frames. A priority-tagged packet has VLAN ID 0 and a CoS value of 0 to 7.

If a packet entering or leaving a port does not match any of the encapsulations on that port, the packet is dropped, resulting in filtering on both ingress and egress. The encapsulation must match the packet on the wire to determine filtering criteria. On the wire refers to packets ingressing the router before any rewrites and to packets egressing the router after all rewrites.

Layer 3 and Layer 4 ACL Support

Beginning in Cisco IOS XE Release 3.3S, support was added for configuring IPv4 Layer 3 and Layer 4 ACLs on EFPs. Configuring an ACL on an EFP is the same as configuring an ACL on other types of interfaces; for example, Ethernet or asynchronous transfer mode (ATM). One exception is that ACLs are not supported for packets prefixed with a Multiprotocol Label Switching (MPLS) header, including when an MPLS packet contains either Layer 3 or Layer 4 headers of supported protocols.

An ACL configured on a main interface containing EFPs does not affect traffic through the EFPs.

To configure an IPv4 Layer 3 and Layer 4 ACL on an EFP, use the **ip access-group** command. An ACL configuration is shown in the "Configuring an ACL on an EFP".

Advanced Frame Manipulation

The Advanced Frame Manipulation feature allows you to specify the VLAN tag manipulation needed on both the incoming and outgoing frames of an EFP. These manipulations include PUSH, POP, and TRANSLATION of one or both VLAN tags.

The PUSH, POP, and TRANSLATION manipulations are as follows:

- PUSH Operations
  - Add one VLAN tag
  - Add two VLAN tags

- POP Operations
    - Remove the outermost VLAN tag
    - Remove the two outermost VLAN tags
- TRANSLATION Operations
    - 1:1 VLAN Translation
    - 1:2 VLAN Translation
    - 2:1 VLAN Translation
    - 2:2 VLAN Translation

When a VLAN tag exists and a new one is added, the CoS field of the new tag is set to the same value as the CoS field of the existing VLAN tag; otherwise, the CoS field is set to a default of 0. Using QoS marking configuration commands, you can change the CoS marking.

EFPs and Layer 2 Protocols

On the Cisco ASR 1000 Series Router, EFPs treat the protocol data units (PDUs) of Layer 2 protocols as data frames. PDUs are forwarded as data frames.

Layer 2 protocols include Cisco Discovery Protocol, Dynamic Trunking Protocol (DTP), Link Aggregation Control Protocol (LACP), Link Layer Discovery Protocol (LLDP), Multiple Spanning Tree Protocol (MSTP), Port Aggregation Protocol (PAgP), Unidirectional Link Detection (UDLD), and VLAN Trunk Protocol (VTP).

Egress Frame Filtering

Egress frame filtering is performed to ensure that frames exiting an EFP contain a Layer 2 header that matches the encapsulation characteristics associated with the EFP. This filtering is done primarily to prevent unintended frame leaks and is always enabled on EFPs.

Bridge Domains

A bridge domain defines a broadcast domain internal to a platform and allows the decoupling of a broadcast domain from a VLAN. This decoupling enables per-port VLAN significance, thus removing the scalability limitations associated with a single per-device VLAN ID space. You can configure a maximum of 4096 EFPs per bridge domain.

A bridge domain interface (BDI) is used to support frame forwarding in a bridge domain at Layer 3. The BDI is a virtual interface that supports Layer 3 features. Each bridge domain can have only one BDI configuration.

If the destination MAC address in a frame received from one of the EFPs participating in a bridge domain matches the BDI MAC address, the frame is routed; otherwise, the frame is bridged. When the egress interface for a routed packet is the BDI interface, the frame is bridged using the destination MAC address.

Frames with broadcast and well-known multicast MAC addresses are also forwarded to the BDI.

The following sections describe support for bridge domains:

EFP, bridge domain, and BDI support based on the Cisco ASR 1000 Series Router forwarding processors are shown in the table in "EFP Bridge Domain and BDI Support Based on the Cisco ASR 1000 Series Router Forwarding Processors".

Ethernet MAC Address Learning

MAC address learning is always enabled and cannot be disabled.

Flooding of Layer 2 Frames for Unknown MAC Multicast and Broadcast Addresses

A Layer 2 frame with an unknown unicast or broadcast destination MAC address is flooded to all the EFPs in the bridge domain except to the originating EFP. A frame with a multicast MAC address is flooded to all the EFPs in the bridge domain. If the destination MAC address is a multicast MAC address, the frame is treated like a broadcast frame and sent to all the EFPs in the bridge domain.

When a frame with either a multicast or broadcast MAC address is flooded and a BDI is associated with the bridge domain, the frame is also flooded to the BDI.

Replication of frames involves recycling the frame several times. This recycling may have a negative effect on forwarding performance and reduce the packet forwarding rate for all features.

Layer 2 Destination MAC Address-Based Forwarding

When bridging is configured, a unicast frame received from an EFP is forwarded based on the destination Layer 2 MAC address. If the destination address is known, the frame is forwarded only to the EFP associated with the destination address.

Because bridge and EFP configurations are interrelated, bridging is supported only on EFPs. To support multiple bridge domains, MAC address entries are associated with the bridge domain of the EFP. Only unicast MAC addresses need to be dynamically learned.

EVC infrastructure does not modify frame contents. Each bridge domain can learn 1000 MAC addresses per second. The Layer 2 frame forwarding rate is 8 million packets per second (MPPS) if flooding is not involved.

MAC Address Aging

The dynamically learned MAC address entries in the MAC table are periodically aged out and entries that are inactive for longer than the configured time period are removed from the table. The supported range of aging-time values, in seconds, is 30 to 600 with a granularity of 1. The default is 5 minutes.

The **aging-time** parameter can be configured per bridge domain and is a relative value. The value is the aging time relative to the time a frame was received with that MAC address.

MAC Address Move

As stations (systems connected to the Cisco ASR 1000 Series Router through the EFP interface) move from one network to another, the interface associated with a MAC address changes.

MAC Address Table

The MAC address table is used to forward frames based on Layer 2 destination MAC addresses. The table consists of static MAC addresses downloaded from the route processor (RP) and the MAC addresses dynamically learned by the data path.

While the MAC Learning feature is enabled, an entry is added to the MAC table when a new unique MAC address is learned on the data path and an entry is deleted from the table when it is aged out.

Split Horizon Group

The split-horizon feature allows service instances in a bridge domain to join groups. Service instances in the same bridge domain and split-horizon group cannot pass data to each other but can forward data to other service instances that are in the same bridge domain and not in the same split-horizon group.

A service instance cannot join more than one split-horizon group. A service instance does not have to be in a split-horizon group. When a service instance does not belong to a group, it can send and receive data from all ports within the bridge domain.

One or more EFPs in a bridge domain may be configured for the same split horizon group, but when a frame is replicated to EFPs, that frame cannot be replicated to EFPs that are within the same split horizon group as the input interface. This restriction applies to MAC address frames that are either known or unknown unicast, broadcast, and multicast frames.

Two split horizon groups per bridge domain are supported on the Cisco ASR 1000 Series Router. You can configure a split horizon group using the **bridge-domain** CLI command with the **split-horizon** and **group** keywords. The group ID can be either 0 or 1.

All members of the bridge-domain that are configured with the same group ID are part of the same split-horizon group. EFPs that are not configured with an explicit group ID do not belong to any group.

EFP Bridge Domain and BDI Support Based on the Cisco ASR 1000 Series Router Forwarding Processors

The table below shows EFP, bridge domain, and BDI support based on the Cisco ASR 1000 Series Router forwarding processors.

Table 4: EFP, Bridge Domain, and BDI Support on the Cisco ASR 1000 Series Router Forwarding Processors

Description	ASR1000-ESP5, ASR 1001, ASR 1002-F (ESP2.5)	ASR1000-ESP10, ASR1000-ESP10-N, ASR1000-ESP20,	ASR1000-ESP40
Maximum EFPs per router	8192	16384	24576
Maximum EFPs per bridge domain	4000	4000	4000
Maximum EFPs per interface	8000	8000	8000
Maximum bridge domains per router	4096	4096	4096
Maximum BDIs per router	4096	4096	4096
Maximum MAC table entries per router	65536	65536	65536
Maximum MAC table entries per bridge domain	16384	16384	16384
Maximum split horizon groups per bridge domain	2	2	2

How to Configure EVCs on the Cisco ASR 1000 Series Router

Configuring an EFP and a Bridge Domain on the Cisco ASR 1000 Series Router

Configuring a service instance on a Layer 2 port creates an EFP on which you can configure EVC features. Perform this task to configure an EFP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation** *encapsulation-type vlan-id*
6. **rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric**
7. **bridge-domain** *bridge-id*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/1/1`	Enters interface configuration mode. • The example shows how to configure Gigabit Ethernet interface 0/1/1 and enter interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode. • The example shows how to configure Ethernet service instance 1.
Step 5	**encapsulation** *encapsulation-type vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 1`	Defines the encapsulation type. • The example shows how to define dot1q as the encapsulation type.

	Command or Action	Purpose
Step 6	**rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric** **Example:** `Router(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 1 symmetric`	(Optional) Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance. • The example shows how to specify translating a single tag defined by the **encapsulation** command to a single tag defined in the **rewrite ingress tag** command with reciprocal adjustment to be done in the egress direction.
Step 7	**bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 1`	Configures the bridge domain. • The example shows how to configure bridge domain 1.

Step 8	**end**
	Example:
	`Router(config-if-srv)# end`
		Returns to privileged EXEC mode.

Configuring an ACL on an EFP

Perform this task to configure an ACL on an EFP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip access-group** *access-list-number* | *access-list-name*} {**in** | **out**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Enters interface configuration mode. • The example shows how to configure Gigabit Ethernet interface 0/1/1 and enter interface configuration mode.

	Command or Action	Purpose
	`Router(config)# interface gigabitethernet 0/1/1`	

Step 4	**ip access-group** *access-list-number* | *access-list-name*} {**in** | **out**}
	Example:
	`Router(config-if)# ip access-group acl55 in`
		Applies an IP access list or object group access control list (OGACL) to an interface or a service policy map.
		• The example shows how to configure an ACL named acl55 for inbound packets.

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

Configuration Examples for EVCs on the Cisco ASR 1000 Series Router

Example Configuring EFPs on a Gigabit Ethernet Interface

```
interface GigabitEthernet0/0/1
 no ip address
 negotiation auto
 service instance 1 ethernet
  encapsulation dot1q 201
  rewrite ingress tag translate 1-to-1 dot1q 300 symmetric
  bridge-domain 1
 !
 service instance 2 ethernet
  encapsulation default
  bridge-domain 1
 !
 service instance 3 ethernet
  encapsulation priority-tagged
  bridge-domain 2
 !
```

Additional References

Related Documents

Related Topic	Document Title
IEEE CFM	"Configuring IEEE Standard-Compliant Ethernet CFM in a Service Provider Network"

Related Topic	Document Title
Using OAM	"Using Ethernet Operations, Administration, and Maintenance"
IEEE CFM and Y.1731 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*

Standards

Standard	Title
IEEE 802.1ag	*802.1ag - Connectivity Fault Management*
IEEE 802.3ah	*Ethernet in the First Mile*
ITU-T	*ITU-T Y.1731 OAM Mechanisms for Ethernet-Based Networks*

Technical Assistance

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

Feature Information for Configuring EVCs on the Cisco ASR 1000 Series Router

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

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

Table 5: Feature Information for Configuring EVCs on the Cisco ASR 1000 Series Router

Feature Name	Releases	Feature Information
ASR1000 EVC Infrastructure	Cisco IOS XE Release 3.2S Cisco IOS XE Release 3.3S	EVC infrastructure is a Layer 2 platform-independent bridging architecture that supports Ethernet services. In Cisco IOS XE Release 3.2S, this feature was introduced on the Cisco ASR 1000 Series Router. The following commands are introduced or modified:**rewrite egress tag**, **rewrite ingress tag**, and **shutdown** (bdomain).
ASR1000 BD Infrastructure	Cisco IOS XE Release 3.2S	Bridge domain infrastructure is a Layer 2 platform-independent architecture that enables bridging. In Cisco IOS XE Release 3.2S this feature was introduced on the Cisco ASR 1000 Series Router. The following sections provide information about support for this feature: The following commands are introduced or modified:**bridge-domain** (service instance), **mac aging-time**.
ACL and QoS Enhancements to EVC Infrastructure in Cisco IOS XE Software	Cisco IOS XE Release 3.3S	Support for configuring Layer 3 and Layer 4 ACLs on EFPs was added in Cisco IOS XE Release 3.3S. The following commands are introduced or modified:**ip access-group**.

CHAPTER 6

Network Interface Device Support

The Network Interface Device (NID) support feature enables support for the NID functionality on a device without including an NID hardware in the network.

Information About NID Support

Network Interface Device Support on the L3 Interface

The Network Interface Device (NID) support feature enables support for the NID functionality on a device without including an NID hardware in the network. This feature combines the Customer-Premises Equipment (CPE) and the NID functionality into a physical device. The following are the advantages of configuring the NID functionality:

- Eliminates the need for a physical NID device and trunk roll.
- Supports both the managed CPE feature set and the NID requirements.

Supported Platforms

The NID Support feature is supported on the following platforms:

- Cisco ISR 4000 Series Integrated Services Routers

Restrictions for NID Support

- Port-channel and EVC interface are not supported .

How to Configure NID Support

Configuring NID Support

Perform the following task to configure NID support:

```
enable
    configure terminal
        interface gigabitEthernet 0/0/2
            no ip address
            port tagging
            encapsulation dot1q 10
            set cos 6
            end
```

Configuration Examples for NID Support

Example: Configuring NID

This configuration example shows how to configure the NID:

```
Device>enable
Device#configure terminal
Device(config)#interface gigabitethernet 0/2
Device(config-if)#port-tagging
Device(config-if-port-tagging)#encapsulation dot1q 10
Device(config-if-port-tagging)#set cos 6
Device(config-if-port-tagging)#end
```

Example: Verifying NID Configuration

Use the following commands to verify the port tagging sessions:

- **show run int**
- **ping**

Use the **show run int** command to display the port tagging sessions:

```
Device#show run interface GigabitEthernet 0/2
Building configuration...
Current configuration : 10585 bytes
!
interface GigabitEthernet0/2
 no ip address
 duplex auto
 speed auto
 port-tagging
  encapsulation dot1q 10
  set cos 6
  exit
end
!
```

```
interface GigabitEthernet0/2.1101
encapsulation dot1Q 100
ip address 10.0.2.4 255.255.255.0
!
interface GigabitEthernet0/2.1102
encapsulation dot1Q 100
ip address 10.0.3.4 255.255.255.0
!
```

Use the **ping** command to verify the connectivity with port tagging configured:

```
Device#ping
 10.0.2.3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.2.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
router#
```

Troubleshooting the NID Configuration

Caution Because debugging output is assigned high priority in the CPU process, it can diminish the performance of the router or even render it unusable. For this reason, use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff.

Note Before you run any of the debug commands listed in the following table, ensure that you run the **logging buffered debugging** command, and then turn off console debug logging using the **no logging console** command.

Table 6: debug Commands for NID Configuration

debug Command	Purpose
debug ethernet nid configuration	Enables debugging of configuration-related issues.
debug ethernet nid packet egress	Enables debugging of packet processing (VLAN tag push) on the egress side.
debug ethernet nid packet ingress	Enables debugging of packet processing (VLAN tag pop) on the ingress side.

Feature Information for NID Support

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

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

Table 7: Feature Information for NID Support

Feature Name	Releases	Feature Information
NID Support	Cisco IOS XE Everest 16.6.1.	The Network Interface Device Support features enables support for the NID functionality on the router without including a NID hardware in the network. No new commands were introduced or modified.

CHAPTER 7

Ethernet Performance Monitoring on Untagged EFPs

The Ethernet Performance Monitoring on untagged EFPs feature enables sessions to run on untagged Ethernet flow points (EFPs).

Information about Ethernet Performance Monitoring on Untagged EFPs

Untagged EFPs

The Ethernet Performance Monitoring on untagged EFPs feature enables sessions to run on untagged Ethernet flow points (EFPs). If an EFP is configured as untagged, then the EFP handles any frames without a dot1q tag, that it receives. Any frames sent using this EFP do not have a dot1q tag.

The dot1q tag contains class of service (CoS) bits, which are used by EPM to test delay or loss of packets with a specific CoS. This support is unavailable when using EPM over untagged EFPs but all other performance monitoring functionality is supported.

How to Configure Ethernet Performance Monitoring on Untagged EFPs

Configuring Ethernet Performance Monitoring on Untagged EFPs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type/number*
4. **service instance** *ID* **ethernet***evc-id*
5. **encapsulation untagged**
6. **end**
7. **configure terminal**
8. **ip sla** *operation-number*
9. **ethernet y1731** {**delay** | **loss**} *type* **domain** *domain-name* {**evc** *evc-id* | **vlan** *vlan-id*} {**mpid** *target-mp-id* | *mac-address target-address*} **cos** *cos-value* {**source** {**mpid** *source-mp-id* | *mac-address tsource-address*}}
10. **exit**
11. **ip sla schedule** *operation-number* **start-time** *time* **life** *life*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type/number* **Example:** `Device(config)# interface GigabitEthernet0/0`	Configures an interface and enters interface configuration mode.
Step 4	**service instance** *ID* **ethernet***evc-id* **Example:** `Device(config-if)# service instance 1 ethernet 50`	Configures a service instance and enters service instance configuration mode.

	Command or Action	Purpose
Step 5	**encapsulation untagged** **Example:** `Device(config-if-srv)# encapsulation untagged`	Sets the encapsulation as untagged.

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

Step 7	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 8	**ip sla** *operation-number* **Example:** `Device(config)# ip sla 501`	Configures a Cisco IOS IP Service Level Agreements (SLAs) operation and enter IP SLA configuration mode.

Step 9	**ethernet y1731** {**delay** | **loss**} *type* **domain** *domain-name* {**evc** *evc-id* | **vlan** *vlan-id*} {**mpid** *target-mp-id* | *mac-address target-address*} **cos** *cos-value* {**source** {**mpid** *source-mp-id* | *mac-address tsource-address*}}
	Example:
	`Device (config-ip-sla)# ethernet y1731 delay DMM domain domain1 evc evc1 mpid 101 cos 0 source mpid 100`
		Begins configuring the receiver on the responder and enters IP SLA Y.1731 delay configuration mode.
		• The source-mp-id or source-address configured by this command corresponds to that of the MEP being configured.
		Note The type argument in the above command syntax takes the following values: DMM, SLM.

Step 10	**exit** **Example:** `Device (config-ip-sla)# exit`	Exits IP SLA configuration mode and returns to privileged EXEC mode.
Step 11	**ip sla schedule** *operation-number* **start-time** *time* **life** *life* **Example:** `Device(config-sla-y1731-delay)# ip sla schedule 501 start-time now life forever`	Begins a probe with a specified operation number starting at the specified timestamp (or 'now' for immediately) for the specified lifetime in seconds (or 'forever' to run until the configuration is removed).

Step 12	**end**
	Example:
	`Device(config-sla-y1731-delay)# end`
		Returns to privileged EXEC mode.

Verifying Ethernet Performance Monitoring on Untagged EFPs

Perform the following task to verify the Ethernet Performance Monitoring on Untagged EFPs

SUMMARY STEPS

1. Enter the **show ip sla statistics** to display performance monitoring sessions with untagged EFPs.

DETAILED STEPS

Enter the **show ip sla statistics** to display performance monitoring sessions with untagged EFPs.

Example:

```
Device# show ip sla statistics
IPSLAs Latest Operation Statistics

IPSLA operation id: 5
Loss Statistics for Y1731 Operation 5
Type of operation: Y1731 Loss Measurement
Latest operation start time: *09:08:29.825 PST Wed Jun 11 2014
Latest operation return code: OK
Distribution Statistics:

Interval
 Start time:  *09:08:29.825 PST Wed Jun 11 2014
 Elapsed time: 9 seconds
 Number of measurements initiated: 8
 Number of measurements completed: 8
 Flag: OK
```

Example for Configuring Ethernet Performance Monitoring on Untagged EFPs

Example: Example for Configuring EPM Untagged EFPs

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet0/0
Device(config-if)# service instance 1 ethernet
Device(config-if-srv)# encapsulation untagged
Device(config-if-srv)# end
Device# configure terminal
Device(config)# ip sla 501
Device(config-ip-sla)# ethernet y1731 delay DMM domain domain1 evc evc1 mpid 101 cos 0
source mpid 100
Device(config-sla-y1731-delay)# exit
Device(config)# ip sla schedule 501 start-time now life forever
Device(config)# end
```

Additional References for Ethernet Performance Monitoring on Untagged EFPs

Related Documents

Related Topic	Document Title
Carrier Ethernet Command Reference	*Cisco IOS Carrier Ethernet Command Reference*
Configuring Ethernet connectivity fault management in a service provider network (Cisco pre-Standard CFM Draft 1)	"Configuring Ethernet Connectivity Fault Management in a Service Provider Network" module in the *Cisco IOS Carrier Ethernet Configuration Guide*
IP SLAs for Metro Ethernet	"IP SLAs for Metro Ethernet"

Technical Assistance

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

Feature Information for Ethernet Performance Monitoring on Untagged EFPs

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

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

Table 8: Feature Information for Ethernet Performance Monitoring on Untagged EFPs

Feature Name	Releases	Feature Information
Ethernet Performance Monitoring on Untagged EFPs	Cisco IOS Release15.5(2)S	The Ethernet Performance Monitoring on untagged EFPs feature enables sessions to run on untagged Ethernet flow points (EFPs). This feature is enabled on Cisco Aggregation Services ASR 903 Series Routers. No commands were introduced or modified.

CHAPTER 8

Using the IEEE 802.3ad Link Aggregation MIB

The IEEE 802.3ad Link Aggregation Control Protocol (LACP) enables the bundling of physical interfaces on a physical device to achieve more bandwidth than is available using a single interface. This feature introduces IEEE 802.3ad Link Aggregation (LAG) MIB support in Cisco IOS XE software. The LAG MIB supports the management of interfaces and ports that are part of an LACP port channel and is accessed by a Simple Network Management Protocol (SNMP) manager application.

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.

Prerequisites for Using the IEEE 802.3ad Link Aggregation MIB

- Link aggregation must be configured using the LACP command-line interface (CLI) commands before the MIB tables can be accessed.
- LACP must be supported in the image.

Information About Using the IEEE 802.3ad Link Aggregation MIB

IEEE 802.3ad LAG MIB

The IEEE 802.3ad LAG MIB supports the management of interfaces and ports that are part of an LACP port channel. No specific commands are used to enable the MIB; access is through an SNMP manager application. For information about using SNMP in Cisco IOS XE software, see the “Configuring SNMP Support” chapter of the *Cisco IOS XE Network Management Configuration Guide*, Release 2.

Configuration and Management of LACP bundles

To use the LAG MIB, it is important to know how LACP bundles are configured and managed. For more information about LACP bundles, see the "Configuring IEEE 802.3ad Link Bundling" feature guide.

LAG MIB Table Object Definitions

This section lists the MIB objects and tables that are supported as part of this feature.

dot3adTablesLastChanged Object

The dot3adTablesLastChanged object indicates the time of the most recent change to the dot3adAggTable, dot3adAggPortListTable, or dot3adAggPortTable.

dot3adAggTable

The dot3adAggTable (Aggregator Configuration table) contains information about every aggregator that is associated with a system. Each LACP channel in a device occupies an entry in the dot3adAggTable. Some objects in the table have restrictions, which are described with the object. The objects are described in the table below.

Table 9: Aggregator Configuration Table Objects

Object	Maximum Access/Description
dot3adAggActorAdminKey	Cannot be changed via the SET operation.
dot3adAggActorOperKey	Write access not supported.
dot3adAggActorSystemID	Write access not supported.
dot3adAggActorSystemPriority	Write access not supported.
dot3adAggAggregateOrIndividual	Returns a value of TRUE if more than 1 port is configured in the channel; otherwise, returns a value of FALSE.
dot3adAggCollectorMaxDelay	Cannot be changed via the SET operation.
dot3adAggIndex	Write access not supported.

Object	Maximum Access/Description
dot3adAggMACAddress	Write access not supported.
dot3adAggPartnerOperKey	Write access not supported.
dot3adAggPartnerSystemID	Write access not supported.
dot3adAggPartnerSystemPriority	Write access not supported.

dot3adAggPortListTable

The dot3adAggPortListTable (Aggregation Port List table) contains a list of all the ports associated with each aggregator. Each LACP channel in a device occupies an entry in the table. The objects are described in the table below.

Table 10: Aggregation Port List Table Objects

Object	Maximum Access/Description
dot3adAggPortListPorts	Write access not supported.

dot3adAggPortTable

The dot3adAggPortTable (Aggregation Port table) contains LACP configuration information about every aggregation port associated with a device. Each physical port in a device occupies an entry in the dot3adAggPortTable. The objects are described in the table below.

Table 11: Aggregation Port Table Objects

Object	Maximum Access/Description
dot3adAggPortActorAdminKey	Write access not supported.
dot3adAggPortActorAdminState	Write access not supported.
dot3adAggPortActorOperKey	Read-only access supported.
dot3adAggPortActorOperState	Write access not supported.
dot3adAggPortActorPort	Write access not supported.
dot3adAggPortActorPortPriority	Write access not supported.
dot3adAggPortActorSystemID	Write access not supported.
dot3adAggPortActorSystemPriority	Write access not supported.
dot3adAggPortAggregateOrIndividual	Indicates whether a port is attached to an LACP channel. If the port is attached to an LACP channel and the value of the dot3adAggPortAttachedAggID object in the same row is not zero, the value of this object is TRUE. Otherwise, the value is FALSE.

Object	Maximum Access/Description
dot3adAggPortAttachedAggID	Write access not supported.
dot3adAggPortIndex	Write access not supported.
dot3adAggPortPartnerAdminKey	Cannot be changed via the SET operation.
dot3adAggPortPartnerAdminPort	Cannot be changed via the SET operation.
dot3adAggPortPartnerAdminPortPriority	Write access not supported.
dot3adAggPortPartnerAdminState	Cannot be changed via the SET operation.
dot3adAggPortPartnerAdminSystemID	Cannot be changed via the SET operation.
dot3adAggPortPartnerAdminSystemPriority	Cannot be changed via the SET operation.
dot3adAggPortPartnerOperKey	Write access not supported.
dot3adAggPortPartnerOperPort	Cannot be changed via the SET operation.
dot3adAggPortPartnerOperPortPriority	Write access not supported.
dot3adAggPortPartnerOperState	Write access is not supported.
dot3adAggPortPartnerOperSystemID	Write access not supported.
dot3adAggPortPartnerOperSystemPriority	Write access not supported.
dot3adAggPortSelectedAggID	Write access not supported.

dot3adAggPortStatsTable

The dot3adAggPortStatsTable (LACP Statistics table) contains link aggregation information about every port that is associated with a device. Each physical port occupies a row in the table. The objects are described in the table below.

Table 12: LACP Statistics Table Objects

Object	Maximum Access/Description
dot3adAggPortStatsIllegalRx	Write access not supported.
dot3adAggPortStatsLACPDUsRx	Write access not supported.
dot3adAggPortStatsLACPDUsTx	Write access not supported.
dot3adAggPortStatsMarkerPDUsRx	Write access not supported.
dot3adAggPortStatsMarkerPDUsTx	Write access not supported.
dot3adAggPortStatsMarkerResponsePDUsRx	Write access not supported.
dot3adAggPortStatsMarkerResponsePDUsTx	Write access not supported.

Object	Maximum Access/Description
dot3adAggPortStatsUnknownRx	Write access not supported.

Additional References

Related Documents

Related Topic	Document Title
Link aggregation configuration tasks	"Configuring IEEE 802.3ad Link Bundling" feature guide
Cisco IOS XE LACP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*
Configuring SNMP	“Configuring SNMP Support” chapter of the *Cisco IOS XE Network Management Configuration Guide*, Release 2
Cisco IOS XE SNMP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Network Management Command Reference*

Standards

Standard	Title
IEEE 802.3ad	*IEEE 802.3ad-2000 Link Aggregation*

MIBs

MIB	MIBs Link
• IEEE 802.3ad MIB • IF 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
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature	--

Technical Assistance

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

Feature Information for Using the IEEE 802.3ad Link Aggregation MIB

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

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

Table 13: Feature Information for Using the IEEE 802.3ad Link Aggregation MIB

Feature Name	Releases	Feature Information
IEEE 802.3ad MIB	Cisco IOS XE Release 2.5	This feature introduces LAG MIB support in Cisco IOS XE software. The LAG MIB supports the management of interfaces and ports that are part of an LACP port channel and is accessed by an SNMP manager application. This feature uses no commands.

CHAPTER 9

Configuring IEEE 802.3ad Link Bundling

This document describes how the IEEE 802.3ad Link Bundling feature leverages the EtherChannel infrastructure within Cisco IOS XE software to manage the bundling of Ethernet links. The supported Ethernet link types for link bundling are Gigabit Ethernet and Ten Gigabit Ethernet.

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.

Prerequisites for Configuring IEEE 802.3ad Link Bundling

- Knowledge of how EtherChannels and Link Aggregation Control Protocol (LACP) function in a network
- Verification that both ends of the LACP link have the same baseline software version

Restrictions for Configuring IEEE 802.3ad Link Bundling

- All links must operate at the same link speed and in full-duplex mode (LACP does not support half-duplex mode).

- EVCs must be with configured **untagged** encapsulation along with L2PT peer, to activate the LACP neighbor configuration.
- All links must be configured as either EtherChannel links or LACP links.
- Only physical interfaces can form aggregations. Aggregations of VLAN interfaces are not possible nor is an aggregation of aggregations.
- If a router is connected to a switch, the bundle terminates on the switch.
- An EtherChannel will not form if one of the LAN ports is a Switched Port Analyzer (SPAN) destination port.
- All ports in an EtherChannel must use the same EtherChannel protocol.
- Maximum of four bundled ports per Ethernet port channel are supported.
- The maximum number of bundled ports per Ethernet port channel that can be supported varies by platform. Some platforms support 4, 8, and 14 while other platforms support a maximum of 16.
- Maximum of 64 Ethernet port channels in a chassis are supported.
- For RSP3, a maximum of 48 Ether channel and a maximum of 8 member-link per Ether channel are supported prior to the Cisco IOS XE Gibraltar 16.11.x release. Starting from the Cisco IOS XE Gibraltar 16.11.x release, 16 member-link per port channel is supported. The restrictions for 8 member-link port channel are also applicable for 16 member-link port channel.
- Quality of service (QoS) is supported on individual bundled ports and not on Ethernet port channels.
- Generic Routing Encapsulation (GRE) is not supported.
- Media type should be uniform across 1G and 10G links.
- For load balancing across 16 member links per port channel, a wide range of addresses (such as Source MAC, Destination MAC, Source IP, Destination IP, and VC) should be used to have the traffic flowing across all the16 member links.

- Quality of service (QoS) is supported on individual bundled ports and not on Ethernet port channels.
- Generic Routing Encapsulation (GRE) is not supported.
- Media type should be uniform across 1G and 10G links.
- For load balancing across 16 member links per port channel, a wide range of addresses (such as Source MAC, Destination MAC, Source IP, Destination IP, and VC) should be used to have the traffic flowing across all the16 member links.

- LACP neighbor comes up on dot1q tagged EFP. This is a known behavior.
- Effective with Cisco IOS XE Fuji 16.9.1, the micro-BFD enabled with port channel having minimum link set to the total member links, is not supported.

Information About Configuring IEEE 802.3ad Link Bundling

Gigabit EtherChannel

Gigabit EtherChannel (GEC) is high-performance Ethernet technology that provides Gigabit per second (Gb/s) transmission rates. A Gigabit EtherChannel bundles individual Ethernet links (Gigabit Ethernet or Ten Gigabit Ethernet) into a single logical link that provides the aggregate bandwidth of up to physical links. All LAN ports in each EtherChannel must be the same speed and all must be configured as either Layer 2 or Layer 3 LAN ports. Inbound broadcast and multicast packets on one link in an EtherChannel are blocked from returning on any other link in the EtherChannel.

When a link within an EtherChannel fails, traffic previously carried over the failed link switches to the remaining links within that EtherChannel. Also when a failure occurs, a trap is sent that identifies the device, the EtherChannel, and the failed link.

Port-Channel and LACP-Enabled Interfaces

Each EtherChannel has a numbered port-channel interface that must be manually created before interfaces can be added to the channel group. The configuration of a port-channel interface affects all LAN ports assigned to that port-channel interface.

To change the parameters of all ports in an EtherChannel, change the configuration of the port-channel interface; for example, if you want to configure Spanning Tree Protocol or configure a Layer 2 EtherChannel as a trunk. Any configuration or attribute changes you make to the port-channel interface are propagated to all interfaces within the same channel group as the port-channel; that is, configuration changes are propagated to the physical interfaces that are not part of the port-channel but are part of the channel group.

The configuration of a LAN port affects only that LAN port.

IEEE 802.3ad Link Bundling

The IEEE 802.3ad Link Bundling feature provides a method for aggregating multiple Ethernet links into a single logical channel based on the IEEE 802.3ad standard. This feature helps improve the cost effectiveness of a device by increasing cumulative bandwidth without necessarily requiring hardware upgrades. In addition, IEEE 802.3ad Link Bundling provides a capability to dynamically provision, manage, and monitor various aggregated links and enables interoperability between various Cisco devices and devices of third-party vendors.

LACP supports the automatic creation of EtherChannels by exchanging LACP packets between LAN ports. LACP packets are exchanged only between ports in passive and active modes. The protocol “learns” the capabilities of LAN port groups dynamically and informs the other LAN ports. After LACP identifies correctly matched Ethernet links, it facilitates grouping the links into an EtherChannel. Then the EtherChannel is added to the spanning tree as a single bridge port.

Both the passive and active modes allow LACP to negotiate between LAN ports to determine if they can form an EtherChannel, based on criteria such as port speed and trunking state. (Layer 2 EtherChannels also use VLAN numbers.) LAN ports can form an EtherChannel when they are in compatible LACP modes, as in the following examples:

- A LAN port in active mode can form an EtherChannel with another LAN port that is in active mode.
- A LAN port in active mode can form an EtherChannel with another LAN port in passive mode.

- A LAN port in passive mode cannot form an EtherChannel with another LAN port that is also in passive mode because neither port will initiate negotiation.

LACP uses the following parameters:

- LACP system priority—You must configure an LACP system priority on each device running LACP. The system priority can be configured automatically or through the command-line interface (CLI). LACP uses the system priority with the device MAC address to form the system ID and also during negotiation with other systems.
- LACP port priority—You must configure an LACP port priority on each port configured to use LACP. The port priority can be configured automatically or through the CLI. LACP uses the port priority to decide which ports should be put in standby mode when there is a hardware limitation that prevents all compatible ports from aggregating. LACP also uses the port priority with the port number to form the port identifier.
- LACP administrative key—LACP automatically configures an administrative key value on each port configured to use LACP. The administrative key defines the ability of a port to aggregate with other ports. A port’s ability to aggregate with other ports is determined by the following:
    - Port physical characteristics such as data rate, duplex capability, and point-to-point or shared medium
    - Configuration restrictions that you establish

On ports configured to use LACP, it tries to configure the maximum number of compatible ports in an EtherChannel, up to the maximum allowed by the hardware. To use the hot standby feature in the event a channel port fails, both ends of the LACP bundle must support the **lacp max-bundle** command.

As a control protocol, LACP uses the Slow Protocol Multicast address of 01-80-C2-00-00-02 to transmit LACP protocol data units (PDUs). Aside from LACP, the Slow Protocol linktype is to be utilized by operations, administration, and maintenance (OAM) packets, too. Subsequently, a subtype field is defined per the IEEE 802.3ad standard [1] (Annex 43B, section 4) differentiating LACP PDUs from OAM PDUs.

Note LACP and Port Aggregation Control Protocol (PAgP) are not compatible. Ports configured for PAgP cannot form port channels on ports configured for LACP, and ports configured for LACP cannot form port channels on ports configured for PAgP.

Benefits of IEEE 802.3ad Link Bundling

- Increased network capacity without changing physical connections or upgrading hardware
- Cost savings from the use of existing hardware and software for additional functions
- A standard solution that enables interoperability of network devices
- Port redundancy without user intervention when an operational port fails

LACP Enhancements

The following LACP enhancements are supported:

- Four member links per LACP bundle.

- Cisco nonstop forwarding (NSF), and nonstop routing (NSR) on Gigabit EtherChannel bundles.
- Link failover time of 250 milliseconds or less and a maximum link failover time of 2 seconds; port channels remain in the LINK_UP state to eliminate reconvergence by the Spanning-Tree Protocol.
- Shutting down a port channel when the number of active links falls below the minimum threshold. In the port channel interface, a configurable option is provided to bring down the port channel interface when the number of active links falls below the minimum threshold. For the port-channel state to be symmetric on both sides of the channel, the peer must also be running LACP and have the same **lacp min-bundle** command setting.
- The IEEE Link Aggregation Group (LAG) MIB.

LACP for Gigabit Interfaces

The LACP (802.3ad) for Gigabit Interfaces feature bundles individual Ethernet links (Gigabit Ethernet or Ten Gigabit Ethernet) into a single logical link that provides the aggregate bandwidth of up to four physical links.

All LAN ports on a port channel must be the same speed and must all be configured as either Layer 2 or Layer 3 LAN ports. If a segment within a port channel fails, traffic previously carried over the failed link switches to the remaining segments within the port channel. Inbound broadcast and multicast packets on one segment in a port channel are blocked from returning on any other segment of the port channel.

Note The network device may impose its own limits on the number of bundled ports per port channel.

Features Supported on Gigabit EtherChannel Bundles

The table below lists the features that are supported on Gigabit EtherChannel (GEC) bundles.

Table 14: Gigabit EtherChannel Bundle Features

Cisco IOS XE Release	Feature	Bundle Interface
2.5	Access control lists (ACLs) per bundle	Supported
	All Ethernet routing protocols	Supported
	Intelligent Service Gateway (ISG) IP sessions	Not Supported
	Interface statistics	Supported
	IP switching	Supported
	IPv4: unicast and multicast	Supported
	IPv6: unicast without load balancing across member links	Supported
	IPv6: multicast	
	Layer 2 Tunneling Protocol Version 3 (L2TPv3), IPinIP, Any Transport Over Multiprotocol Label Switching (MPLS) (AToM) tunnels	Supported
	Layer 2 Tunneling Protocol Version 2 (L2TPv2)	Not Supported
	MPLS (6PE)	Supported
	Multicast VPN	Not Supported
	VLANs	Supported
2.6	Virtual Private Network (VPN) Routing and Forwarding (VRF)	Supported
3.4	IPv6: unicast and multicast	Supported
3.6	Bidirectional Forwarding Detection (BFD) over GEC	Supported
3.7	Layer 2 Tunneling Protocol Version 2 (L2TPv2)	Supported
	PPPoX (PPPoEoE, PPPoEoQinQ, PPPoVLAN)	Supported
3.7.6	Policy-based routing (PBR) over GEC	Supported
3.11	GEC over L2TPv3	Supported

Cisco IOS XE Release	Feature	Bundle Interface
3.12	MPLS TE (Traffic Engineering) over GEC	Supported

Guidelines for LACP for Gigabit Interfaces Configuration

Port channel interfaces that are configured improperly with LACP are disabled automatically to avoid network loops and other problems. To avoid configuration problems, observe these guidelines and restrictions:

- Every port added to a port channel must be configured identically. No individual differences in configuration are allowed.
- Bundled ports can be configured on different line cards in a chassis.
- Maximum transmission units (MTUs) must be configured on only port channel interfaces; MTUs are propagated to the bundled ports.
- QoS and committed access rate (CAR) are applied at the port level. Access control lists (ACLs) are applied on port channels.
- MAC configuration is allowed only on port channels.
- MPLS IP should be enabled on bundled ports using the **mpls ip** command.
- Unicast Reverse Path Forwarding (uRPF) should be applied on the port channel interface using the **ip verify unicast reverse-path** command in interface configuration mode.
- Cisco Discovery Protocol should be enabled on the port channel interface using the **cdp enable** command in interface configuration mode.
- All LAN ports in a port channel should be enabled. If you shut down a LAN port in a port channel, the shutdown is treated as a link failure and the traffic is transferred to one of the remaining ports in the port channel.
- Create a port channel interface using the **interface port-channel** command in global configuration mode.
- When an Ethernet interface has an IP address assigned, disable that IP address before adding the interface to the port channel. To disable an existing IP address, use the **no ip address** command in interface configuration mode.
- The **hold queue in** command is valid only on port channel interfaces. The **hold queue out** command is valid only on bundled ports.

How to Configure IEEE 802.3ad Link Bundling

Enabling LACP

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface port-channel** *channel-number*
4. **channel-group** *channel-group-number* **mode** {**active** | **passive**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:** `Device(config)# interface port-channel 10`	Identifies the interface port channel and enters interface configuration mode.

Step 4	**channel-group** *channel-group-number* **mode** {**active** | **passive**}
	Example:
	`Device(config-if)# channel-group 25 mode active`
		Configures the interface in a channel group and sets it as active.
		In active mode, the port will initiate negotiations with other ports by sending LACP packets.

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

Configuring a Port Channel

You must manually create a port channel logical interface. Perform this task to configure a port channel.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *channel-number*
4. **lacp max-bundle** *max-bundles*
5. **ip address** *ip-address mask*
6. **end**
7. **show running-config interface port-channel** *group-number*

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:** `Device(config)# interface port-channel 10`	Identifies the interface port channel and enters interface configuration mode.
Step 4	**lacp max-bundle** *max-bundles* **Example:** `Device(config-if)# lacp max-bundle 3`	Configures three active links on the port channel. The remaining links are in standby mode. Traffic is load-balanced among the active links.
Step 5	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.31.52.10 255.255.255.0`	Assigns an IP address and subnet mask to the EtherChannel.

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

Step 7	**show running-config interface port-channel** *group-number* **Example:** `Device# show running-config interface port-channel 10`	Displays the port channel configuration.

Example

This example shows how to verify the configuration:

```
Device# show running-config interface port-channel 10

Building configuration...
Current configuration: : 110 bytes
!
interface Port-channel10
ip address 172.31.52.10 255.255.255.0
no negotiation auto
lacp max-bundle 3
end
```

Configuring LACP (802.3ad) for Gigabit Interfaces

Perform this task to create a port channel with two bundled ports. You can configure a maximum of four bundled ports per port channel.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *number*
4. **ip address** *ip-address mask*
5. **interface** *type slot*/*subslot*/ *port*
6. **no ip address**
7. **channel-group** *channel-group-number* **mode** {**active** | **passive**}
8. **exit**
9. **interface** *type slot*/*subslot*/ *port*
10. **no ip address**
11. **channel-group** *channel-group-number* **mode** {**active** | **passive**}
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *number* **Example:** `Device(config)# interface port-channel 1`	Specifies the port channel interface and enters interface configuration mode. • *number* —Valid range is from 1 to 64.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.1.1.1 255.255.255.0`	Assigns an IP address and subnet mask to the port channel interface.
Step 5	**interface** *type slot*/*subslot*/ *port* **Example:** `Device(config-if)# interface gigabitethernet`	Specifies the port to bundle.
Step 6	**no ip address** **Example:**	Disables the IP address on the port channel interface.

	Command or Action	Purpose
	`Device(config-if)# no ip address`	

Step 7	**channel-group** *channel-group-number* **mode** {**active** | **passive**}
	Example:
	`Device(config-if)# channel-group 1 mode active`
		Assigns the interface to a port channel group and sets the LACP mode.
		• *channel-group-number* —Valid range is 1 to 64.
		• **active** —Places a port into an active negotiating state, in which the port initiates negotiations with other ports by sending LACP packets.
		• **passive** —Places a port into a passive negotiating state, in which the port responds to LACP packets it receives but does not initiate LACP negotiation. In this mode, the channel group attaches the interface to the bundle.

Step 8	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 9	**interface** *type* *slot*/*subslot*/ *port* **Example:** `Device(config)# interface gigabitethernet`	Specifies the next port to bundle and places the CLI in interface configuration mode.
Step 10	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables the IP address on the port channel interface.

Step 11	**channel-group** *channel-group-number* **mode** {**active** | **passive**}
	Example:
	`Device(config-if)# channel-group 1 mode active`
		Assigns the interface to the previously configured port channel group.
		• *channel-group-number* —Valid range is 1 to 64.
		• **active** —Places a port into an active negotiating state, in which the port initiates negotiations with other ports by sending LACP packets.
		• **passive** —Places a port into a passive negotiating state, in which the port responds to LACP packets it receives but does not initiate LACP negotiation. In this mode, the channel-group attaches the interface to the bundle.

Step 12	**end**
	Example:
	`Device(config-if)# end`
		Returns to privileged EXEC mode.

Example

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 1
Device(config-if)# ip address 10.1.1.1 255.255.255.0
Device(config-if)#
Device(config-if)# no ip address
Device(config-if)# channel-group 1 mode active
Device(config-if)# exit
Device(config)#
Device(config-if)# no ip address
Device(config-if)# channel-group 1 mode active
Device(config-if)# end
```

Setting LACP System Priority and Port Priority

Perform this task to set the LACP system priority and port priority. The system ID is the combination of the LACP system priority and the MAC address of a device. The port identifier is the combination of the port priority and port number.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **lacp system-priority** *priority*
4. **interface** *slot*/*subslot*/ *port*
5. **lacp port-priority** *priority*
6. **end**
7. **show lacp sys-id**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**lacp system-priority** *priority* **Example:** `Device(config)# lacp system-priority 200`	Sets the system priority.

	Command or Action	Purpose
Step 4	**interface** *slot*/*subslot*/ *port* **Example:** `Device(config)# interface gigabitethernet 0/1/1`	Specifies the bundled port on which to set the LACP port priority and enters interface configuration mode.
Step 5	**lacp port-priority** *priority* **Example:** `Device(config-if)# lacp port-priority 500`	Specifies the priority for the physical interface. • *priority* —Valid range is from 1 to 65535. The higher the number, the lower the priority.

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

Step 7	**show lacp** **sys-id** **Example:** `Device# show lacp sys-id`	Displays the system ID (a combination of the system priority and the MAC address of the device).

Examples

```
Device> enable
Device# configure terminal
Device(config)# lacp system-priority 200
Device(config)# interface gigabitethernet 0/1/1
Device(config-if)# lacp port-priority 500
Device(config-if)# end
```

This example shows how to verify the LACP configuration:

```
Device# show lacp sys-id
200.abdc.abcd.abcd
```

Adding and Removing Interfaces from a Link Bundle

SUMMARY STEPS

1. **enable**
2. **configure** **terminal**
3. **interface** *type* *slot*/*subslot*/*port*
4. **channel-group** *channel-group-number* **mode** {**active** | **passive**}
5. **no channel-group** *channel-group-number* **mode** {**active** | **passive**}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot*/*subslot*/*port* **Example:** `Device(config)# interface gigabitethernet`	Configures a Gigabit Ethernet interface.

Step 4	**channel-group** *channel-group-number* **mode** {**active** | **passive**}
	Example:
	`Device(config-if)# channel-group 5 mode active`
		Adds an interface to a channel group and enters interface configuration mode.
		• In this instance, the interface from Step 3 is added.

Step 5	**no channel-group** *channel-group-number* **mode** {**active** | **passive**}
	Example:
	`Device(config-if)# no channel-group 5 mode active`
		Removes the Gigabit Ethernet interface from channel group.

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

Removing a Channel Group from a Port

Perform this task to remove a Gigabit Ethernet port channel group from a physical port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no interface port-channel** *number*
4. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no interface port-channel** *number* **Example:** `Device(config)# no interface port-channel 1`	Removes the specified port channel group from a physical port.

Step 4	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Example

```
Device> enable
Device# configure terminal
Device(config)# no interface port-channel 1
Device(config)# end
```

Setting a Minimum Threshold of Active Links

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **lacp min-bundle** *min-bundle*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Creates a port-channel virtual interface and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface port-channel 1`	
Step 4	**lacp min-bundle** *min-bundle* **Example:** `Device(config-if)# lacp min-bundle 1`	Sets the minimum threshold of active links to 1. **Note** For Cisco ASR 1000 Series Aggregation Services Routers, the minimum number of member links per GEC interface is 1 and the maximum number is 14.

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

Monitoring LACP Status

SUMMARY STEPS

1. **enable**
2. **show lacp** {*number* | **counters** | **internal** | **neighbor** | **sys-id**}

DETAILED STEPS

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

Step 2	**show lacp** {*number* | **counters** | **internal** | **neighbor** | **sys-id**}
	Example:
	`Device# show lacp internal`
		Displays internal device information.

Troubleshooting Tips

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

1. Check the device error status.
2. When a error exists, perform a loopback test to confirm the error.
3. Run a traceroute to the destination to isolate the fault.
4. If the fault is identified, correct the fault.
5. If the fault is not identified, go to the next lower maintenance domain and repeat steps 1 through 4 at that maintenance domain level.

6. Repeat the first four steps, as needed, to identify and correct the fault.

Displaying Gigabit EtherChannel Information

To display Gigabit Ethernet port channel information, use the **show interfaces port-channel** command in user EXEC mode or privileged EXEC mode. The following example shows information about port channels configured on ports 0/2 and 0/3. The default MTU is set to 1500 bytes.

```
Device# show interfaces port-channel 1
Port-channel1 is up, line protocol is up
Hardware is GEChannel, address is 0013.19b3.7748 (bia 0000.0000.0000)
MTU 1500 bytes, BW 2000000 Kbit, DLY 10 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
ARP type: ARPA, ARP Timeout 04:00:00
No. of active members in this channel: 2
Member 0 : GigabitEthernet , Full-duplex, 1000Mb/s Member 1 : GigabitEthernet , Full-duplex,
 1000Mb/s
Last input 00:00:05, output never, output hang never
Last clearing of "show interface" counters 00:04:40
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
Interface Port-channel1 queueing strategy: PXF First-In-First-Out
Output queue 0/8192, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts (0 IP multicasts)
0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
0 watchdog, 0 multicast, 0 pause input
3 packets output, 180 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 babbles, 0 late collision, 0 deferred
0 lost carrier, 0 no carrier, 0 PAUSE output
0 output buffer failures, 0 output buffers swapped out
```

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

*Table 15: **show interfaces port-channel** Field Descriptions*

Field	Description
Port-channel1 is up, line protocol is up	Indicates the bundle interface is currently active and can transmit and receive or it has been taken down by an administrator.
Hardware is	Hardware type (Gigabit EtherChannel).
address is	Address being used by the interface.
MTU	Maximum transmission unit of the interface.
BW	Bandwidth of the interface, in kilobits per second.
DLY	Delay of the interface, in microseconds.
reliability	Reliability of the interface as a fraction of 255 (255/255 is 100 percent reliability), calculated as an exponential average over 5 minutes.

Field	Description
tx load rxload	Transmit and receive load on the interface as a fraction of 255 (255/255 is completely saturated), calculated as an exponential average over 5 minutes. The calculation uses the value from the **bandwidth** interface configuration command.
Encapsulation	Encapsulation type assigned to the interface.
loopback	Indicates if loopbacks are set.
keepalive	Indicates if keepalives are set.
ARP type	Address Resolution Protocol (ARP) type on the interface.
ARP Timeout	Number of hours, minutes, and seconds an ARP cache entry stays in the cache.
No. of active members in this channel	Number of bundled ports (members) currently active and part of the port channel group.
Member <*no.*> Gigabit Ethernet: <*no.* /*no.* /*no.* >	Number of the bundled port and associated Gigabit Ethernet port channel interface.
Last input	Number of hours, minutes, and seconds since the last packet was successfully received by an interface and processed locally on the Device. Useful for knowing when a dead interface failed. This counter is updated only when packets are process-switched, not when packets are fast-switched.
output	Number of hours, minutes, and seconds since the last packet was successfully transmitted by an interface. This counter is updated only when packets are process-switched, not when packets are fast-switched.
output hang	Number of hours, minutes, and seconds since the interface was last reset because of a transmission that took too long. When the number of hours in any of the “last” fields exceeds 24 hours, the number of days and hours is printed. If that field overflows, asterisks are printed.
last clearing	Time at which the counters that measure cumulative statistics (such as number of bytes transmitted and received) shown in this report were last reset to zero. Variables that might affect routing (for example, load and reliability) are not cleared when the counters are cleared. *** indicates that the elapsed time is too long to be displayed. 0:00:00 indicates that the counters were cleared more than 231 ms and less than 232 ms ago.
Input queue	Number of packets in the input queue and the maximum size of the queue.
Queueing strategy	First-in, first-out queueing strategy (other queueing strategies you might see are priority-list, custom-list, and weighted fair).
Output queue	Number of packets in the output queue and the maximum size of the queue.
5 minute input rate 5 minute output rate	Average number of bits and packets received or transmitted per second in the last 5 minutes.

Field	Description
packets input	Total number of error-free packets received by the system.
bytes (input)	Total number of bytes, including data and MAC encapsulation, in the error-free packets received by the system.
no buffer	Number of received packets discarded because there was no buffer space in the main system. Broadcast storms on Ethernet lines and bursts of noise on serial lines are often responsible for no input buffer events.
broadcasts	Total number of broadcast or multicast packets received by the interface.
runts	Number of packets that are discarded because they are smaller than the minimum packet size for the medium.
giants	Number of packets that are discarded because they exceed the maximum packet size for the medium.
input errors	Total number of no buffer, runts, giants, cyclic redundancy checks (CRCs), frame, overrun, ignored, and terminated counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts.
CRC	CRC generated by the originating LAN station or far-end device does not match the checksum calculated from the data received. On a LAN, this usually indicates noise or transmission problems on the LAN interface or the LAN bus. A high number of CRCs is usually the result of collisions or a station transmitting bad data. On a serial link, CRCs usually indicate noise, gain hits or other transmission problems on the data link.
frame	Number of packets received incorrectly having a CRC error and a noninteger number of octets. On a serial line, this is usually the result of noise or other transmission problems.
overrun	Number of times the serial receiver hardware was unable to pass received data to a hardware buffer because the input rate exceeded the receiver's capacity for handling the data.
ignored	Number of received packets ignored by the interface because the interface hardware ran low on internal buffers. These buffers are different than the system buffers mentioned previously in the buffer description. Broadcast storms and bursts of noise can cause the ignored count to be incremented.
watchdog	Number of times the watchdog receive timer expired.
multicast	Number of multicast packets received.
packets output	Total number of messages transmitted by the system.
bytes (output)	Total number of bytes, including data and MAC encapsulation, transmitted by the system.
underruns	Number of times that the far-end transmitter has been running faster than the near-end Device's receiver can handle.

Field	Description
output errors	Sum of all errors that prevented the final transmission of datagrams out of the interface being examined. Note that this might not balance with the sum of the enumerated output errors, as some datagrams can have more than one error, and others can have errors that do not fall into any of the specifically tabulated categories.
collisions	Number of messages retransmitted because of an Ethernet collision. A packet that collides is counted only once in output packets.
interface resets	Number of times an interface has been completely reset. This can happen if packets queued for transmission were not sent within a certain interval. If the system notices that the carrier detect line of an interface is up but the line protocol is down, the system periodically resets the interface in an effort to restart that interface. Interface resets can also occur when an unrecoverable interface processor error occurred, or when an interface is looped back or shut down.
babbles	The transmit jabber timer expired.
late collision	Number of late collisions. Late collision happens when a collision occurs after transmitting the preamble. The most common cause of late collisions is that your Ethernet cable segments are too long for the speed at which you are transmitting.
deferred	Indicates that the chip had to defer while ready to transmit a frame because the carrier was asserted.
lost carrier	Number of times the carrier was lost during transmission.
no carrier	Number of times the carrier was not present during the transmission.
PAUSE output	Not supported.
output buffer failures	Number of times that a packet was not output from the output hold queue because of a shortage of shared memory.
output buffers swapped out	Number of packets stored in main memory when the output queue is full; swapping buffers to main memory prevents packets from being dropped when output is congested. The number is high when traffic is bursty.

Configuration Examples for IEEE 802.3ad Link Bundling

Example: Configuring LACP for Gigabit Interfaces

The following example shows how to configure Gigabit Ethernet ports into port channel 1 with LACP parameters.

```
Device> enable
Device# configure terminal
Device(config)# lacp system-priority 65535
Device(config)# interface port-channel 1
Device(config-if)# lacp max-bundle 1
Device(config-if)# ip address 10.1.1.1 255.255.255.0
Device(config-if)# exit
Device(config)#
Device(config-if)# no ip address
Device(config-if)# lacp port-priority 100
Device(config-if)# channel-group 1 mode passive
Device(config-if)# exit
Device(config)#
Device(config-if)# no ip address
Device(config-if)# lacp port-priority 200
Device(config-if)# channel-group 1 mode passive
Device(config-if)# end
```

Example Associating a Channel Group with a Port Channel

This example shows how to configure channel group number 5 and include it in the channel group.

```
Device1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device1(config)# interface port 5
Device1(config-if)#
*Aug 20 17:06:14.417: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel5, changed
 state to down
*Aug 20 17:06:25.413: %LINK-3-UPDOWN: Interface Port-channel5, changed state to down
Device1(config-if)#
Device1(config-if)#
Device1(config-if)# channel-group 5 mode active
Device1(config-if)#
*Aug 20 17:07:43.713: %LINK-3-UPDOWN: Interface GigabitEthernet, changed state to down
*Aug 20 17:07:44.713: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet,
changed state to down
*Aug 20 17:07:45.093: %C10K_ALARM-6-INFO: ASSERT CRITICAL GigE  Physical Port Link Down
*Aug 20 17:07:45.093: %C10K_ALARM-6-INFO: CLEAR CRITICAL GigE  Physical Port Link Down
*Aug 20 17:07:47.093: %LINK-3-UPDOWN: Interface GigabitEthernet, changed state to up
*Aug 20 17:07:48.093: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet,
changed state to up
*Aug 20 17:07:48.957: GigabitEthernet added as member-1 to port-channel5

*Aug 20 17:07:51.957: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel5, changed
 state to up
Device1(config-if)# end
Device1#
*Aug 20 17:08:00.933: %SYS-5-CONFIG_I: Configured from console by console
Device1# show etherchannel summary
Flags:  D - down        P/bndl - bundled in port-channel
        I - stand-alone s/susp - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

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


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

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1       Po1(RU)         LACP     Te0/3/0(bndl) Te0/3/1(hot-sby)
```

```
RU - L3 port-channel UP State
SU - L2 port-channel UP state
P/bndl -  Bundled
S/susp  - Suspended

Device1# show running-config int po1
Building configuration...

Current configuration : 87 bytes
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
end

Device1# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
   SA      bndl      32768          0x5       0x5     0x43        0x3D
Device1# show interface port 5
Port-channel5 is up, line protocol is up
  Hardware is GEChannel, address is 0014.a93d.4aa8 (bia 0000.0000.0000)
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
    No. of active members in this channel: 1
        Member 0 : GigabitEthernet , Full-duplex, 1000Mb/s
  Last input 00:00:05, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Interface Port-channel5 queueing strategy: PXF First-In-First-Out
  Output queue 0/8192, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     9 packets output, 924 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 PAUSE output
     0 output buffer failures, 0 output buffers swapped out
```

Example Adding and Removing Interfaces from a Bundle

The following example shows how to add an interface to a bundle:

```
Device1# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5
                            LACP port     Admin     Oper    Port        Port
```

```
Port      Flags   State     Priority      Key       Key     Number      State
   SA      bndl      32768         0x5       0x5     0x43        0x3D
Device1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device1(config)#
Device1(config-if)# channel-group 5 mode active
Device1(config-if)#
*Aug 20 17:10:19.057: %LINK-3-UPDOWN: Interface GigabitEthernet, changed state to down
*Aug 20 17:10:19.469: %C10K_ALARM-6-INFO: ASSERT CRITICAL GigE  Physical Port Link Down
*Aug 20 17:10:19.473: %C10K_ALARM-6-INFO: CLEAR CRITICAL GigE  Physical Port Link Down
*Aug 20 17:10:21.473: %LINK-3-UPDOWN: Interface GigabitEthernet, changed state to up
*Aug 20 17:10:21.473: GigabitEthernet taken out of port-channel5
*Aug 20 17:10:23.413: GigabitEthernet added as member-1 to port-channel5

*Aug 20 17:10:23.473: %LINK-3-UPDOWN: Interface Port-channel5, changed state to up
Device1(config-if)# end
Device1#
*Aug 20 17:10:27.653: %SYS-5-CONFIG_I: Configured from console by console
*Aug 20 17:11:40.717: GigabitEthernet added as member-2 to port-channel5

Device1# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
   SA      bndl      32768         0x5       0x5     0x43        0x3D
   SA      bndl      32768         0x5       0x5     0x42        0x3D
Device1#
Device1# show interface port 5
Port-channel5 is up, line protocol is up
  Hardware is GEChannel, address is 0014.a93d.4aa8 (bia 0000.0000.0000)
  MTU 1500 bytes, BW 2000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
    No. of active members in this channel: 2
        Member 0 : GigabitEthernet , Full-duplex, 1000Mb/s  <---- added to port channel
bundle
        Member 1 : GigabitEthernet , Full-duplex, 1000Mb/s
  Last input 00:00:00, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/150/0/0 (size/max/drops/flushes); Total output drops: 0
  Interface Port-channel5 queueing strategy: PXF First-In-First-Out
  Output queue 0/8192, 0 drops; input queue 0/150, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     104 packets output, 8544 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 PAUSE output
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how to remove an interface from a bundle:

```
Device1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device1(config)#
```

```
Device1(config-if)# no channel-group 5 mode active
Device1(config-if)#
*Aug 20 17:15:49.433: GigabitEthernet taken out of port-channel5
*Aug 20 17:15:49.557: %C10K_ALARM-6-INFO: ASSERT CRITICAL GigE  Physical Port Link Down
*Aug 20 17:15:50.161: %C10K_ALARM-6-INFO: CLEAR CRITICAL GigE  Physical Port Link Down
*Aug 20 17:15:51.433: %LINK-3-UPDOWN: Interface GigabitEthernet, changed state to down
*Aug 20 17:15:52.433: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet,
changed state to down
Device1(config-if)# end
Device1#
*Aug 20 17:15:58.209: %SYS-5-CONFIG_I: Configured from console by console
Device1#
*Aug 20 17:15:59.257: %C10K_ALARM-6-INFO: ASSERT CRITICAL GigE  Physical Port Link Down
*Aug 20 17:15:59.257: %C10K_ALARM-6-INFO: CLEAR CRITICAL GigE  Physical Port Link Down
Device1#
*Aug 20 17:16:01.257: %LINK-3-UPDOWN: Interface GigabitEthernet, changed state to up
*Aug 20 17:16:02.257: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet,
changed state to up
Device1# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
   SA      bndl      32768         0x5       0x5     0x42        0x3D
```

Example Monitoring LACP Status

The following example shows LACP activity that you can monitor by using the **show lacp** command.

```
Device1# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
   SA      bndl      32768         0x5       0x5     0x42        0x3D
Device1# show lacp 5 counters
             LACPDUs         Marker      Marker Response    LACPDUs
Port       Sent   Recv     Sent   Recv     Sent   Recv      Pkts Err
---------------------------------------------------------------------
Channel group: 5
     21     18       0      0        0      0         0
Device1# show lacp 5 internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
   SA      bndl      32768         0x5       0x5     0x42        0x3D
Device1# show lacp 5 neighbor
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 5 neighbors
Partner's information:
          Partner Partner   LACP Partner  Partner   Partner  Partner     Partner
Port      Flags   State     Port Priority Admin Key Oper Key Port Number Port State
   SP      32768     0011.2026.7300  11s    0x1     0x14     0x3C
Device1# show lacp counters
```

```
             LACPDUs         Marker      Marker Response     LACPDUs
Port       Sent   Recv     Sent   Recv     Sent   Recv      Pkts Err
---------------------------------------------------------------------
Channel group: 5
     23     20       0      0        0      0         0
Device1# show lacp sys-id
32768,0014.a93d.4a00
```

Example: Displaying Port-Channel Interface Information

The following example shows how to display the configuration of port-channel interface 1.

```
Device# show interface port-channel 1
Port-channel1 is up, line protocol is up
Hardware is GEChannel, address is 0013.19b3.7748 (bia 0000.0000.0000)
MTU 1500 bytes, BW 2000000 Kbit, DLY 10 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
ARP type: ARPA, ARP Timeout 04:00:00
No. of active members in this channel: 2
Member 0 : GigabitEthernet , Full-duplex, 1000Mb/s Member 1 : GigabitEthernet , Full-duplex,
 1000Mb/s
Last input 00:00:05, output never, output hang never
Last clearing of "show interface" counters 00:04:40
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
Interface Port-channel1 queueing strategy: PXF First-In-First-Out
Output queue 0/8192, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts (0 IP multicasts)
0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
0 watchdog, 0 multicast, 0 pause input
3 packets output, 180 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 babbles, 0 late collision, 0 deferred
0 lost carrier, 0 no carrier, 0 PAUSE output
0 output buffer failures, 0 output buffers swapped out
```

Additional References Configuring IEEE 802.3ad Link Bundling

Related Documents

Related Topic	Document Title
Configuring EtherChannels	"Configuring Layer 3 and Layer 2 EtherChannel" chapter of the *Catalyst 6500 Release 12.2SXF Software Configuration Guide*
LACP commands	Cisco IOS Carrier Ethernet Command Reference
LACP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS Network Management Command Reference

Standards

Standard	Title
IEEE 802.3ad-2000	*IEEE 802.3ad-2000 Link Aggregation*

Technical Assistance

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

Feature Information for Configuring IEEE 802.3ad Link Bundling

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

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

Table 16: Feature Information for Configuring IEEE 802.3ad Link Bundling

Feature Name	Releases	Feature Information
EtherChannel Min-Links	Cisco IOS XE Release 2.5 Cisco IOS XE Release 3.8S	The EtherChannel Min-Links feature allows a port channel to be shut down when the number of active links falls below the minimum threshold. Using the **lacp min-bundle** command, you can configure the minimum threshold. In Cisco IOS XE Release 3.8S, support was added for the Cisco ASR 903 Router. The following commands were introduced or modified: **lacp min-bundle**.
IEEE 802.3ad Faster Link Switchover Time	Cisco IOS XE Release 2.5	The IEEE 802.3ad Faster Link Switchover Time feature provides a link failover time of 250 milliseconds or less and a maximum link failover time of 2 seconds. Also, port channels remain in the LINK_UP state to eliminate reconvergence by the Spanning-Tree Protocol. The following commands were introduced or modified: **lacp fast-switchover**.

Feature Name	Releases	Feature Information
IEEE 802.3ad Link Aggregation (LACP)	Cisco IOS XE Release 2.4	The IEEE 802.3ad Link Aggregation feature provides a method for aggregating multiple Ethernet links into a single logical channel based on the IEEE 802.3ad standard. In addition, this feature provides a capability to dynamically provision, manage, and monitor various aggregated links and enables interoperability between various Cisco devices and devices of third-party vendors. In Cisco IOS XE Release 2.4, this feature was implemented on the Cisco ASR1000 Series Router. The following commands were introduced or modified: **channel-group (interface)**, **debug lacp**, **lacp max-bundle**, **lacp port-priority**, **lacp system-priority**, **show lacp**.
Link Aggregation Control Protocol (LACP) (802.3ad) for Gigabit Interfaces	Cisco IOS XE Release 2.5	The LACP (802.3ad) for Gigabit Interfaces feature bundles individual Gigabit Ethernet links into a single logical link that provides the aggregate bandwidth of up to four physical links. The following commands were introduced or modified: **lacp max-bundle**.
SSO - LACP	Cisco IOS XE Release 2.5	The SSO - LACP feature supports stateful switchover (SSO), in service software upgrade (ISSU), Cisco nonstop forwarding (NSF), and nonstop routing (NSR) on Gigabit EtherChannel bundles. This feature uses no new or modified commands.
Support for 14 Member-links per GEC Bundle	Cisco IOS XE Denali 16.3.1	The Support for 14 Member-links per GEC Bundle feature extends the number of supported member links from 4 to 14 on Cisco ASR 1000 Series Aggregation Services Routers. This feature uses no new or modified commands.

CHAPTER 10

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.

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.

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

- IEEE-compliant connectivity fault management (CFM) must be configured and enabled for Y.1731 performance monitoring to function.

Note Y1731 is supported over Port Channel interfaces.

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
- Drop eligibility—marked drop ineligible
- Transmission rate
- Total interval of ETH-DM
- MEF10 frame-delay variation algorithm

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:

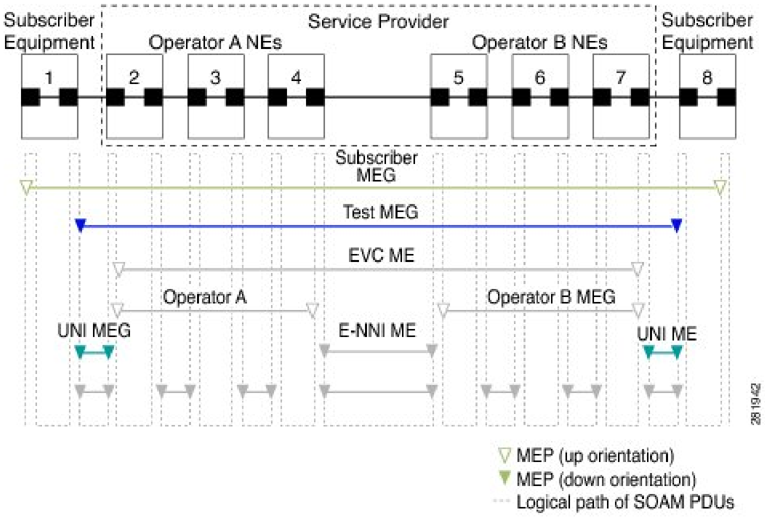
- RxTimeStampf—Time stamp of the time at which the frame with ETH-DM request information was received.
- TxTimeStampb—Time stamp of the time at which the transmitting frame with ETH-DM reply information was sent.
- The timestamping happens at the hardware level for DMM operations.

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 3: Y.1731 Performance Monitoring



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 Performance Monitoring Parameters

The following new commands were introduced that can be used to configure and display performance monitoring parameters: **debug ethernet cfm pm**, **monitor loss counters**, and **show ethernet cfm pm**.

For more information about CFM and Y.1731 performance monitoring commands, see the *Cisco IOS Carrier Ethernet Command Reference*. For more information about debug commands, see the *Cisco IOS Debug Command Reference*.

Configuration Examples for Configuring ITU-T Y.1731 Performance Monitoring Functions

Example: Configuring Performance Monitoring

For Y.1731 performance monitoring configuration examples, see Configuring IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations. For information on Y.1731 On-Demand and Concurrent Operations see, IPSLA Y1731 On-Demand and Concurrent Operations.

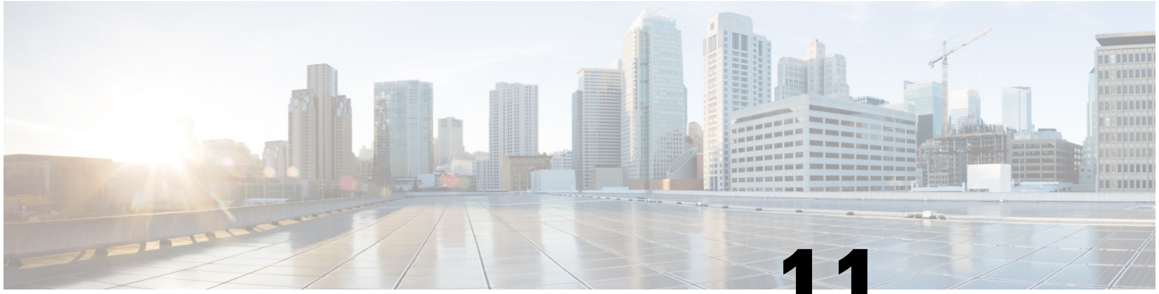
Feature Information for ITU-T Y.1731 Performance Monitoring in a Service Provider Network

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

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

Table 17: Feature Information for ITU-T Y.1731 Performance Monitoring in a Service Provider Network

Feature Name	Releases	Feature Information
Y.1731 Performance Monitoring	Cisco IOS XE Release 3.5S	The Y.1731 Performance Monitoring feature describes the Ethernet performance monitoring aspect of SLAs such as frame loss, frame delay, and frame-delay variation. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router. In Cisco IOS XE Release 3.6S, support for port channels and cross connect functionality was provided. The following commands were introduced or modified: **debug ethernet cfm pm**, **ethernet cfm distribution enable**, **monitor loss counters**, **show ethernet cfm pm**.



CHAPTER 11

Enabling Ethernet Local Management Interface

Ethernet Local Management Interface (LMI) is an Ethernet layer operation, administration, and management (OAM) protocol. It provides information that enables autoconfiguration of customer edge (CE) devices and provides the status of Ethernet virtual connections (EVCs) for large Ethernet metropolitan-area networks (MANs) and WANs. Specifically, Ethernet LMI 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 and a user-network interface (UNI) to a CE device.

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.

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.

Prerequisites for Enabling Ethernet Local Management Interface

Business Requirements

- Ethernet operation, administration, and management (OAM) such as connectivity fault management (CFM) must be implemented and operational on the service provider’s network.

Restrictions for Enabling Ethernet Local Management Interface

- Ethernet Local Management Interface (LMI) relies on Ethernet connectivity fault management (CFM) for the status of an Ethernet virtual circuit (EVC), the remote user network interface (UNI) identifier associated with an EVC, and remote UNI status.
- Ethernet LMI customer edge (CE) is available only on routing ports on routing platforms. For information about Ethernet LMI provider edge (PE) functionality on switching platforms, see the “Configuring Ethernet CFM and E-LMI” chapter of the *Cisco ME 3400 Switch Software Configuration Guide*.
- Not all Cisco software releases support autoconfiguration of CE devices.

Information About Enabling Ethernet Local Management Interface

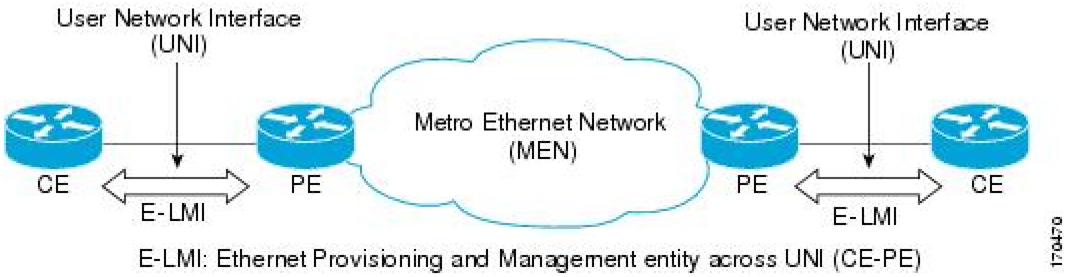
EVC

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

Ethernet LMI

Ethernet Local Management Interface (LMI) is an Ethernet layer operation, administration, and management (OAM) protocol between a customer edge (CE) device and the provider edge (PE) device in large Ethernet MANs and WANs. It provides information that enables service providers to autoconfigure CE devices with service parameters and parameter changes from a user provider edge (UPE) device.

The figure below shows where in a network Ethernet LMI functions.



LMI also provides the status of Ethernet virtual circuits (EVCs) in large Ethernet MANs and WANs to the CE. Specifically, Ethernet LMI 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 EVC and user network identifier (UNI) attributes to a CE device.

The Ethernet LMI protocol includes the following procedures, as defined by the MEF 16 Technical Specification:

- Notifying the CE when an EVC is added
- Notifying the CE when an EVC is deleted
- Notifying the CE of the availability state of a configured EVC (Active, Not Active, or Partially Active)
- Communicating UNI and EVC attributes to the CE

Benefits of Ethernet LMI

- Communication of end-to-end status of the EVC to the CE device
- Communication of EVC and UNI attributes to a CE device
- Competitive advantage for service providers

How to Enable Ethernet Local Management Interface

Enabling Ethernet LMI on All Supported Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet lmi global**
4. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet lmi global** **Example:** `Device(config)# ethernet lmi global`	Enables Ethernet Local Management Interface (LMI) on all supported interfaces on the device.

Step 4	**end**
	Example:
	`Device# end`
		Returns to privileged EXEC mode.

Enabling Ethernet LMI on a Single Supported Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet lmi interface**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ethernet lmi interface** **Example:** `Device(config-if)# ethernet lmi interface`	Enables Ethernet Local Management Interface (LMI) on the interface.

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

Configuration Examples for Ethernet Local Management Interface

The examples in this section show the configurations that enable Ethernet LMI on all interfaces on a CE device (globally) and on a specific interface on a CE device.

Example: Enabling Ethernet LMI on All Supported Interfaces

```
enable
configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ethernet lmi global
end
00:06:33: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/0, changed p
```

Example: Enabling Ethernet LMI on a Single Supported Interface

```
enable
configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
interface ethernet 0/0
ethernet lmi interface
end
00:05:51: %SYS-5-CONFIG_I: Configured from console by console
```

Additional References for Enabling Ethernet Local Management Interface

Related Documents

Related Topic	Document Title
Ethernet Connectivity Fault Management (CFM)	“Configuring Ethernet Connectivity Fault Management in a Service Provider Network” in the *Cisco IOS Carrier Ethernet Configuration Guide*
Configuring CFM and Ethernet Local Management Interface (E-LMI) in a service provider network	*Cisco ME 3400 Switch Software Configuration Guide, Rel. 12.2(25)SEG*
Commands used for configuring Ethernet LMI in a service provider network	*Cisco ME 3400 Switch Command Reference, Rel. 12.2(25)SEG*
Ethernet LMI at a provider edge	“Configuring Ethernet Local Management Interface at a Provider Edge” in the *Carrier Ethernet Configuration Guide*
Carrier Ethernet commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*

Standards

Standard	Title
Metro Ethernet Forum 16 Technical Specification	Technical Specification MEF 16- Ethernet Local Management Interface
IEEE P802.1ag/D5.2	*Draft Standard for Local and Metropolitan Area Networks*
ITU-T Q.3/13	Liaison statement on Ethernet OAM (Y.17ethoam)
IETF VPLS OAM	*L2VPN OAM Requirements and Framework*

Technical Assistance

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

Feature Information for Enabling Ethernet Local Management Interface

Table 18: Feature Information for Enabling Ethernet Local Management Interface

Feature Name	Releases	Feature Information
Ethernet Local Management Interface	Cisco IOS XE Release 3.9S	Ethernet LMI is an Ethernet layer OAM protocol. It provides information that enables autoconfiguration of CE devices and provides the status of EVCs for large Ethernet MANs and WANs. The following commands were introduced or modified: **clear ethernet lmi statistics**, **debug ethernet lmi**, **ethernet lmi**, **ethernet lmi global**, **ethernet lmi interface**, **show ethernet lmi**.

Glossary

CE --customer edge. Edge equipment on the customer side of a user-network interface (UNI).

CE-VLAN ID --Identifier of a CE-VLAN.

E-LMI --Ethernet Local Management Interface. An Ethernet layer OAM protocol. It provides information that enables autoconfiguration of CE devices and provides the status of Ethernet virtual connections (EVCs) for large Ethernet MANs and WANs.

EVC --Ethernet virtual connection. An association of two or more user-network interfaces.

OAM --operations, administration, and maintenance. A term used by several standards bodies to describe protocols and procedures for operating, administrating, and maintaining networks. Examples are ATM OAM and IEEE Std. 802.3ah OAM.

PE --provider edge. Edge equipment on the service provider side of a user-network interface (UNI).

UNI --user-network interface. A common term for the connection point between an operator’s bridge and customer equipment. A UNI often includes a C-VLAN-aware bridge component. The term UNI is used broadly in the IEEE P802.1ag/D5.2 standard when the purpose for various features of LMI are explained.

CHAPTER 12

Layer 2 Access Control Lists on EVCs

The ability to filter packets in a modular and scalable way is important for both network security and network management. Access Control Lists (ACLs) provide the capability to filter packets at a fine granularity. In Metro Ethernet networks, ACLs are directly applied on Ethernet virtual circuits (EVCs).

Layer 2 Access Control Lists on EVCs is a security feature that allows packet filtering based on MAC addresses. This module describes how to implement ACLs on EVCs.

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.

Prerequisites for Layer 2 Access Control Lists on EVCs

- Knowledge of how service instances must be configured.
- Knowledge of extended MAC ACLs and how they must be configured.

Restrictions for Layer 2 Access Control Lists on EVCs

Information About Layer 2 Access Control Lists on EVCs

EVCs

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. It is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer. An EVC contains the different parameters on which the service is being offered. A service instance is the instantiation of an EVC on a specified port.

Service instances are configured under a port channel. The traffic carried by the service instance is load balanced across member links. Service instances under a port channel are grouped and each group is associated with one member link. Ingress traffic for a single EVC can arrive on any member of the bundle. All egress traffic for a service instance uses only one of the member links. Load balancing is achieved by grouping service instances and assigning them to a member link.

Ethernet virtual connection services (EVCS) uses the EVCs and service instances to provide Layer 2 switched Ethernet services. EVC status can be used by a customer edge (CE) device either to find an alternative path 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.

For information about the Metro Ethernet Forum standards, see the Standards table in the “Additional References” section.

Relationship Between ACLs and Ethernet Infrastructure

The following points capture the relationship between ACLs and Ethernet Infrastructure (EI):

- ACLs can be directly applied on an EVC using the command-line interface (CLI). An ACL is applied to a service instance, which is the instantiation of an EVC on a given port.
- One ACL can be applied to more than one service instance at any time.
- One service instance can have one ACL at most applied to it at any time. If a Layer 2 ACL is applied to a service instance that already has a Layer 2 ACL, the new one replaces the old one.
- Only named ACLs can be applied to service instances. The command syntax ACLs is retained; the **mac access-list extended** command is used to create an ACL.
- The command can be used to provide details about ACLs on service instances.

How to Configure Layer 2 Access Control Lists on EVCs

Creating a Layer 2 ACL

Perform this task to create a Layer 2 ACL with a single ACE.

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 **mac access-list extended** *name*

Example:

```
Device(config)# mac access-list extended test-12-acl
```

Defines an extended MAC ACL and enters mac access list control configuration mode.

Step 4 **permit** {{*src-mac mask* | **any**} {*dest-mac mask* | **any**} [*protocol* [**vlan** *vlan*] [*cos value*]]}

Example:

```
Device(config-ext-macl)# permit 00aa.00bb.00cc 0.0.0 any
```

Allows forwarding of Layer 2 traffic if the conditions are matched. Creates an ACE for the ACL.

Applying a Layer 2 ACL to a Service Instance

Perform this task to apply a Layer 2 ACL to a service instance. Note that packet filtering takes place only after the ACL has been created and applied to the service instance.

Before you begin

Before applying an ACL to a service instance, you must create it using the **mac access-list extended command. See the "Creating a Layer 2 ACL" section.**

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* ethernet
5. **encapsulation dot1q** *vlan-id*
6. **mac access-group** *access-list-name* in

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/0/0`	Specifies the type and location of the interface to configure, where: • *type* --Specifies the type of the interface. • *number* --Specifies the location of the interface.
Step 4	**service instance** *id* ethernet **Example:** `Device(config-if)# service instance 100 ethernet`	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**mac access-group** *access-list-name* in **Example:** `Device(config-if-srv)# mac access-group test-12-acl in`	Applies a MAC ACL to control incoming traffic on the interface.

Configuring a Layer 2 ACL with ACEs on a Service Instance

Perform this task to configure the same ACL with three ACEs and stop all other traffic on a service instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mac access-list extended** *name*
4. **permit** {*src-mac mask* | **any**} {*dest-mac mask* | **any**}
5. **permit** {*src-mac mask* | **any**} {*dest-mac mask* | **any**}
6. **permit** {*src-mac mask* | **any**} {*dest-mac mask*} | **any**}
7. **deny any any**
8. **exit**
9. **interface** *type number*
10. **service instance** *id* **ethernet**
11. **encapsulation dot1q** *vlan-id*
12. **mac access-group** *access-list-name* **in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac access-list extended** *name* **Example:** `Device(config)# mac access list extended test-12-acl`	Defines an extended MAC ACL and enters mac access control list configuration mode.

Step 4	**permit** {*src-mac mask* | **any**} {*dest-mac mask* | **any**}
	Example:
	`Device(config-ext-macl)# permit 00aa.bbcc.ddea 0.0.0 any`
		Allows forwarding of Layer 2 traffic if the conditions are matched. This creates an ACE for the ACL.

Step 5	**permit** {*src-mac mask* | **any**} {*dest-mac mask* | **any**}
	Example:
	`Device(config-ext-macl)# permit 00aa.bbcc.ddeb 0.0.0 any`
		Allows forwarding of Layer 2 traffic if the conditions are matched. This creates an ACE for the ACL.

Step 6	**permit** {*src-mac mask* | **any**} {*dest-mac mask*} | **any**} **Example:**	Allows forwarding of Layer 2 traffic if the conditions are matched. This creates an ACE for the ACL.

	Command or Action	Purpose
	`Device(config-ext-macl)# permit 00aa.bbcc.ddec 0.0.0 any`	
Step 7	**deny any any** **Example:** `Device(config-ext-macl)# deny any any`	Prevents forwarding of Layer 2 traffic except for the allowed ACEs.
Step 8	**exit** **Example:** `Device(config-ext-macl)# exit`	Exits the current command mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/0/0`	Specifies the interface.
Step 10	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 200 ethernet`	Configures an Ethernet service instance on an interface and enters service instance configuration mode.
Step 11	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.
Step 12	**mac access-group** *access-list-name* **in** **Example:** `Device(config-if-srv)# mac access-group test-12-acl in`	Applies a MAC ACL to control incoming traffic on the interface.

Verifying the Presence of a Layer 2 ACL on a Service Instance

Perform this task to verify that a Layer 2 ACL is present on an EVC. This verification task can be used after an ACL has been configured to confirm its presence.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type number* detail

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type* *number* detail **Example:** `Device# show ethernet service instance id 100 interface gigabitethernet 3/0/1 detail`	Displays detailed information about Ethernet customer service instances.

Configuration Examples for Layer 2 Access Control Lists on EVCs

Example Applying a Layer 2 ACL to a Service Instance

The following example shows how to apply a Layer 2 ACL called mac-20-acl to a service instance. The ACL has five permitted ACEs and all other traffic is not allowed.

```
enable
configure terminal
 mac access-list extended mac-20-acl

 permit 00aa.bbcc.adec 0.0.0 any

 permit 00aa.bbcc.bdec 0.0.0 any

 permit 00aa.bbcc.cdec 0.0.0 any

 permit 00aa.bbcc.edec 0.0.0 any

 permit 00aa.bbcc.fdec 0.0.0 any

 deny any any
 exit
interface gigabitethernet 10/0/0
 service instance 100 ethernet
 encapsulation dot1q 100
 mac access-group mac-20-acl in
```

Example Applying a Layer 2 ACL to Three Service Instances on the Same Interface

The following example shows how to apply a Layer 2 ACL called mac-07-acl to three service instances on the same interface:

```
enable
configure terminal
mac access-list extended mac-07-acl


permit 00aa.bbcc.adec 0.0.0 any


permit 00aa.bbcc.bdec 0.0.0 any


permit 00aa.bbcc.cdec 0.0.0 any


deny any any
exit
interface gigabitethernet 10/0/0
service instance 100 ethernet
encapsulation dot1q 100
mac access-group mac-07-acl in
service instance 101 ethernet
encapsulation dot1q 101
mac access-group mac-07-acl in
service instance 102 ethernet
encapsulation dot1q 102
mac access-group mac-07-acl in
```

Example Creating a Layer 2 ACL with ACEs

The following example shows how to create a Layer 2 ACL called mac-11-acl with two permitted ACEs:

```
enable
configure terminal
mac access-list extended mac-11-acl
permit 00aa.00bb.00cc 1a11.0101.11c1 any
permit 00aa.00bb.00cc 1a11.0101.11c2 any
```

Example Displaying the Details of a Layer 2 ACL on a Service Instance

The following sample output displays the details of a Layer 2 ACL called test-acl on a service instance.

The table below describes the significant fields in the output.

Table 19: show ethernet service instance Field Descriptions

Field	Description
Service Instance ID	Displays the service instance ID.
L2 ACL (inbound):	Displays the ACL name.

Field	Description
Associated Interface:	Displays the interface details of the service instance.
Associated EVC:	Displays the EVC with which the service instance is associated.
CEVlans:	Displays details of the associated VLAN ID.
State:	Displays whether the service instance is in an up or down state.
L2 ACL permit count:	Displays the number of packet frames allowed to pass on the service instance by the ACL.
L2 ACL deny count	Displays the number of packet frames not permitted to pass on the service instance by the ACL.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS Carrier Ethernet commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*
Configuring CFM over an EFP Interface with the Cross Connect feature on the Cisco ASR 903 Router.	*Configuring the CFM over EFP Interface with Cross Connect Feature*
Configuring Ethernet Virtual Connections on the Cisco ASR 903 Router	*Configuring Ethernet Virtual Connections on the Cisco ASR 903 Router*

Standards

Standard	Title
MEF 6.1	*Metro Ethernet Services Definitions Phase 2 (PDF 6/08)*
MEF 10.1	*Ethernet Services Attributes Phase 2 (PDF 10/06)*

Technical Assistance

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

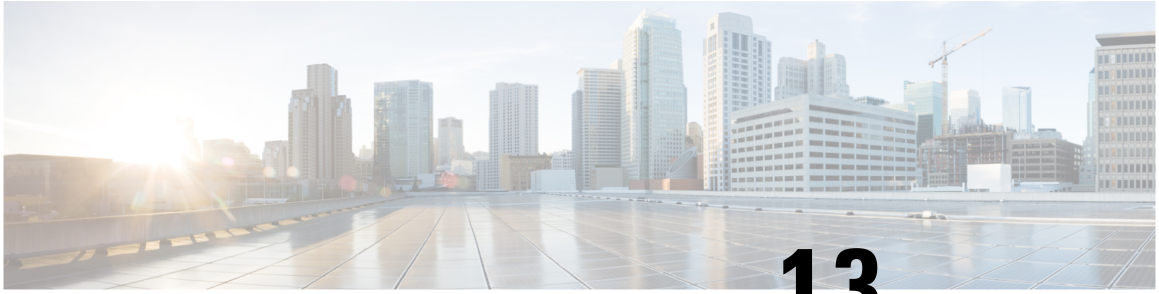
Feature Information for Layer 2 Access Control Lists on EVCs

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

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

Table 20: Feature Information for Layer 2 Access Control Lists on EVCs

Feature Name	Releases	Feature Information
Layer 2 Access Control Lists on EVCs	Cisco IOS XE Release 3.6S	The Layer 2 Access Control Lists on EVCs feature introduces ACLs on EVCs. • The following commands were introduced or modified: **interface, mac access-group in**, **mac access-list extended, show ethernet service instance**.



CHAPTER 13

Layer 2 Ethernet over GRE

Ethernet over Soft Generic Routing Encapsulation (EoGRE) is an aggregation solution for aggregating WiFi traffic from hotspots. This solution enables customer premises equipment (CPE) devices to bridge the Ethernet traffic from an end host, and encapsulates the traffic in Ethernet packets over an IP GRE tunnel. The IP GRE tunnel terminates on a service provider broadband network gateway, which then terminates the end host traffic and manages the subscriber session for the end host.

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 Layer 2 Ethernet over GRE

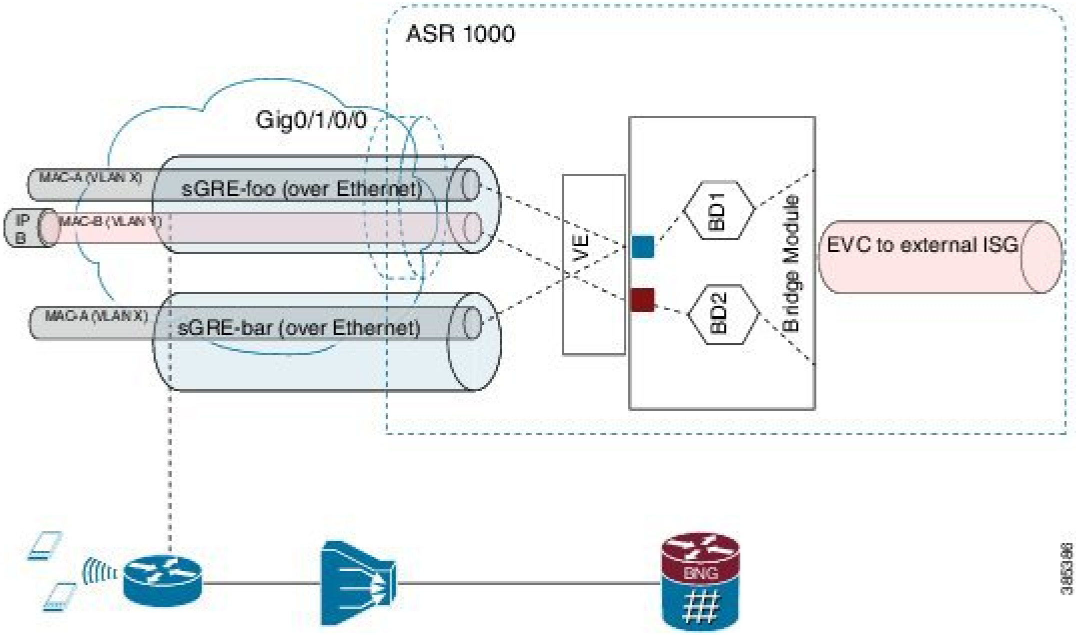
- Transport on IPv6 is not supported.
- Virtual Ethernet interface does not support encapsulation untagged.
- L2 EoGRE is only supported on Cisco ASR 1000 routers. It is not supported on Cisco CSR1000V or Cisco 4000 Series ISR routers.
- Multicast traffic is not supported.

Information About Layer 2 Ethernet over GRE

The ASR 1000 platform services as the SP broadband network gateway which:

- Terminates the IP GRE tunnel, and/or
- Manages the subscriber session for the end-host client.

Figure 4: Ethernet over Soft GRE Deployment



The deployment model that is supported is the two-box model:

- In the two-box model, the ASR 1000 router provides only the functionality to terminate the bridged Ethernet over soft GRE traffic. The ISG subscriber management resides in an external router which is connected with the router using L2 bridge-domain.

The major components involved in L2 EoGRE are:

- Virtual Ethernet interface.
- Ethernet service instance.
- IP GRE tunnel data plane.
- L2 bridge-domain data plane.

Control Plane—After the Virtual Ethernet interface is configured, the Virtual Ethernet interface is downloaded to the ASR 1000 platform as a virtual interface.

Service instances (EVCs) are configured under the virtual ethernet interface and are downloaded to ASR1K platform. The service instances are then propagated in the ASR 1000 platform to fman-rp, fman-fp, cpp-client and cpp dataplane, where the EVC feature invocation arrays (FIAs) are enabled.

Data Plane—When receiving the IP GRE encapsulated Ethernet packet, the data plane tunnel ingress processing checks the protocol field in the GRE header. If the protocol is transparent Ethernet bridging protocol (0x6558), the packet is identified as Ethernet over soft GRE packet and is directed to the Ethernet service instance classification module. The Ethernet service instance classification module classifies the packet into the service instance configured under the Virtual Ethernet interface using the VLAN tag in the packet. After the Ethernet service instance is identified, the packet goes through the programmed processing FIA under the Ethernet service instance such as vlan tag manipulation and is then sent to L2 bridge domain for further processing. At the L2 bridge domain processing module, the client source MAC address is dynamically learned, and so is the IP GRE tunnel end-points. As a result, the MAC address table contains the IP GRE tunnel end-points for the client MAC address.

The packet is then either bridged to the external ISG for subscriber processing in the case of the two-box deployment model. In the direction where the ISG, either external or internal, sends a packet to the mobile client, the L2 bridge domain looks up the destination MAC address using the L2 bridge domain MAC address table. Once the result is found, the IP GRE tunnel end point addresses are also retrieved from the MAC address table. The L2 Ethernet packet is then encapsulated into the IP GRE tunnel using the retrieved tunnel end point address. Once encapsulated into the IP GRE packet, IP lookup is then performed and the packet is sent to the CPE.

Configuration Example: Two-box Deployment Model

```
Interface GigabitEthernet0/0/0
  description Connect-Internet-Gateway
  no ip address
  negotiation auto
  service instance 140 Ethernet
    encapsulation dot1q 140
    bridge-domain 140

Interface Virtual-Ethernet1
    description L2 EoGRE Tunnel
    arp timeout 0
    service instance 140 Ethernet
      encapsulation dot1q 140
      rewrite egress tag translate 1-to-1 dot1q 140
      bridge-domain 140
```

Additional References

Related Documents

Related Topic	Document Title
Configuration guide	*Cisco IOS Carrier Ethernet Configuration Guide*, Release 12.2SR
Carrier Ethernet commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*

Standards

Standard	Title
None	--

MIBs

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

RFCs

RFC	Title
None	--

Technical Assistance

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

Feature Information for Layer 2 Ethernet over GRE

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

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

Table 21: Feature Information for Layer 2 Ethernet over GRE

Feature Name	Releases	Feature Information
Layer 2 Ethernet over GRE	Cisco IOS XE Release 3.9S	Ethernet over Soft Generic Routing Encapsulation (EoGRE) is an aggregation solution for aggregating WiFi traffic from hotspots. This solution enables customer premises equipment (CPE) devices to bridge the Ethernet traffic from an end host, and encapsulates the traffic in Ethernet packets over an IP GRE tunnel. The IP GRE tunnel terminates on a service provider broadband network gateway, which then terminates the end host traffic and manages the subscriber session for the end host.

CHAPTER 14

Configuring MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

The MAC Address Limiting on Service Instances, Bridge Domains, and EVC Port Channels feature addresses port security with service instances by providing the capability to control and filter MAC address learning behavior at the granularity of a per-service instance. When a violation requires a shutdown, only the customer who is assigned to a given service instance is affected and--not all customers who are using the port.

MAC address limiting is a type of MAC security and is also referred to as a MAC security component or element.

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.

Prerequisites for MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

- An understanding of service instances and bridge domains.
- An understanding of how port channels and EtherChannels work in a network.

Restrictions for MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

- MAC address limiting for service instances and bridge domains is configured under a service instance and is permitted only after the service instance is configured under a bridge domain. If a service instance is removed from a bridge domain, all the MAC address limiting commands under it are also removed. If a bridge domain is removed from a service instance, all the MAC address limiting commands are also removed.
- The MAC Address on RSP1 port channel overlaps with the interface MAC address and the traffic is dropped from ports 1 to 8 when an interface module is placed on slot 4.
-
- System wide, the following limits apply to the total configured allowed list and learned MAC addresses:
    - Total number of MAC addresses supported under MAC Security is limited to 64K (65536).
    - Total number of secure EFPs in the system is limited to 64K (65536).
    - Total number of MAC addresses supported under MAC Security, per EFP, is limited to 1K (1024).
    - Total number of EFPs per bridge domain 4000.
    - Total number of bridge domains per system 16000.
- You can configure or remove the various MAC security elements irrespective of whether MAC security is enabled on the EFP. However, these configurations become operational only after MAC security is enabled.
- It is recommended that you enable MAC address security feature on all the EFPs in a bridge-domain.
- When you enable the MAC address security for EVC bridge domain feature, existing MAC address table entries on the EFP are removed.
- When you enable the MAC address security, the traffic is forwarded once the device learns the MAC address.
- The MAC address security for EVC bridge domain feature can be configured on an EFP only if the EFP is a member of a bridge domain.
- you can configure non-MAC address security on an EVC and enable MAC address security on a different EVC, which are in the same bridge-domain.
- If you disassociate the EFP from the BD, the MAC security feature is completely removed.

- For port-channel, this configuration is propagated to all member links in the port-channel. Consistent with the already implemented bridge domain EVC port-channel functionality, packets on a secured EFP are received on any member link, but all the egress packets are sent out to one of the selected member links.
- System does not permit addition of multicast/broadcast MAC address as a permit address. However, addition of multicast/broadcast MAC address is allowed in deny address configuration to verify such invalid packets.
- When EVC with the same EFP or service instance is created between ports 1 and 2 and MAC address m1 is configured as permit address (allowed list) on port1, the same MAC address can be configured as deny address (blocked list) on port2 and vice versa.

Information About MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

Ethernet Virtual Circuits, Service Instances, and Bridge Domains

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. It is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer. An EVC embodies the different parameters on which the service is being offered. A service instance is the instantiation of an EVC on a given port.

Support for Ethernet bridging is an important Layer 2 service that is offered on a router as part of an EVC. Ethernet bridging enables the association of a bridge domain with a service instance.

For information about the Metro Ethernet Forum standards, see the “Standards” table in the “Additional References” section.

EVCs on Port Channels

An EtherChannel bundles individual Ethernet links into a single logical link that provides the aggregate bandwidth of up to eight physical links. The Ethernet Virtual Connection Services (EVCS) EtherChannel feature provides support for EtherChannels on service instances.

Note The MAC Address Security on EVC Port Channel services is supported only on bridge domains over Ethernet and is not supported on xconnect services.

EVCS uses the concepts of EVCs and service instances.

Load balancing is done on an Ethernet flow point (EFP) basis where a number of EFPs exclusively pass traffic through member links.

MAC Security and MAC Addressing

MAC security is enabled on a service instance by configuring the **mac security** command. Various MAC security elements can be configured or removed regardless of whether the **mac security** command is presently configured, but these configurations become operational only when the **mac security** command is applied.

In this document, the term “secured service instance” is used to describe a service instance on which MAC security is configured. The MAC addresses on a service instance on which MAC security is configured are referred to as “secured MAC addresses.” Secured MAC addresses can be either statically configured (as a permit list) or dynamically learned.

MAC Address Permit List

A permit list is a set of MAC addresses that are permitted on a service instance. Permitted addresses permanently configured into the MAC address table of the service instance.

On a service instance that is a member of a bridge domain, the operator is permitted to configure one or more permitted MAC addresses.

For each permitted address, eligibility tests are performed and after the address passes these tests, it is either:

- Programmed into the MAC address table of the bridge domain, if MAC security is enabled on the service instance or,
- Stored in an area of memory referred to as “MAC table cache” if MAC security is not enabled on the service instance. When MAC security is enabled, the addresses from the MAC table cache are added to the MAC address table as secure addresses.

The eligibility tests performed when a user tries to add a MAC address to the permit list on a service instance are as follows:

- If the address is already a denied address on the service instance, the configuration is rejected with an appropriate error message.
- If the acceptance of this address would increase the secure address count on the service instance beyond the maximum number allowed, an attempt is made to make room by removing an existing address from the MAC address table. The only candidate for removal is a dynamically learned address on the service instance. If sufficient room cannot be made, the configuration is rejected. If the acceptance of this address would increase the secure address count on the bridge domain beyond the maximum number allowed, an attempt is made to make room by removing an existing address from the MAC address table. The only candidate for removal is a dynamically learned address on the service instance. If room cannot be made, the configuration is rejected.
- If the address is already permitted on another service instance in the same bridge domain, one of the following actions occur:
    - If the conflicting service instance has MAC security configured, the configuration is rejected with an appropriate error message.
    - If the conflicting service instance does not have MAC security configured, the configuration is accepted silently. (If the operator attempts to enable MAC security on the conflicting service instance, that attempt fails.)

MAC Address Deny List

A deny list is a set of MAC addresses that are not permitted on a service instance. An attempt to learn a denied MAC address will fail. On a service instance that is a member of a bridge domain, the operator is permitted to configure one or more denied MAC addresses. The arrival of a frame with a source MAC address that is part of a deny list will trigger a violation response.

Before a denied address can be configured, the following test is performed:

- If the address is already configured as a permitted address on the specific service instance or if the address has been learned and saved as a sticky address on the service instance, the configuration is rejected with an appropriate error message.

In all other cases, the configuration of the denied address is accepted. Typical cases include:

- The address is configured as a permitted address on another service instance in the same bridge domain, or the address has been learned and saved as a sticky address on another service instance.
- The address is present in the MAC table of the bridge domain as a dynamically learned address on the specific service instance and is deleted from the MAC table before the configuration is accepted.

MAC Address Limiting and Learning

An upper limit for the number of secured MAC addresses allowed on a bridge domain service instance can be configured. This limit includes addresses added as part of a permit list and dynamically learned MAC addresses.

Before an unknown MAC address is learned, a series of checks are run against a set of configured and operational constraints. If any of these checks fails, the address is not learned, and a configured violation response is triggered.

Static and Dynamic MAC Addresses

A static MAC address is specified as permitted on a service instance, by a **mac security permit**command. A dynamic MAC address is a source MAC address encountered by the service instance that is not present in the MAC table but is allowed into and learned by the MAC address table.

Dynamic MAC Address Learning

Dynamic MAC address learning occurs when the bridging data path encounters an ingress frame whose source address is not present in the MAC address table for the ingress secured service instance.

The MAC security component is responsible for permitting or denying the addition of the new source address into the MAC table. The following constraints apply:

- If a MAC address is to be learned, a check is performed to determine whether the number of secured MAC addresses exceed the maximum number that are permitted to be learned on the individual service instance and on the bridge domain.
- A check is performed to determine if the MAC address on an another service instance is learned on a secured service instance in the same bridge domain.
- A check is performed to verify if the new dynamic MAC address is in a deny list.

MAC Address Limiting on Service Instances

The user can configure the maximum number of MAC addresses that can exist in the MAC table that is associated with a service instance. This number includes statically configured and dynamically learned (including sticky) addresses.

On a service instance that has MAC security enabled and that does not have the maximum number of MAC addresses configured, the number of addresses allowed is one. This means that if the service instance has an associated permit list, that permit list can have only one address, and no addresses are learned dynamically. If the service instance does not have an associated permit list, one MAC address may be learned dynamically.

MAC Address Limiting for Bridge Domains

An upper limit for the number of MAC addresses that can reside in the MAC address table of a bridge domain can be set. This is set independently of the upper limit of secured MAC addresses on the service instance. An attempted violation of this bridge domain MAC address limit will cause the MAC address learn attempt to fail, and the frame to be dropped.

If the bridge domain MAC address limit is not configured, then by default, the maximum number of MAC addresses allowed on a bridge domain is the maximum number that can be supported by that platform.

Relationship Between the MAC Address Limit on a Bridge Domain and on a Service Instance

You can specify the maximum count of MAC table entries on a bridge domain and on a service instance simultaneously. However, there are no restrictions on the count that is configured on the service instance.

The table below shows an example of an initial configuration where three service instances are configured on a bridge domain:

Table 22: Bridge-Domain and Service-Instance MAC Address Limit

Bridge-Domain / Service-Instance Number	MAC Address Limit
Bridge Domain 1000	20
Service Instance 1001	5
Service Instance 1002	10
Service Instance 1003	To be configured

If you wish to configure MAC security on service instance 1003, any value can be configured for the maximum count. For example:

```
service instance 1003 ethernet
 bridge-domain 1
 mac security
 mac security maximum addresses 35
```

A MAC address limit of 35 is permitted, even though the total MAC address limit for the three service instances (5 + 10 + 35) would exceed the count (20) configured on the bridge domain. Note that during actual operation, the bridge domain limit of 20 is in effect. The dynamic secure address count cannot exceed the lowest count applicable, so it is not possible for service instance 1003 to learn 35 addresses.

MAC Move and MAC Locking

If a MAC address is present in the MAC address table for a service instance (for example, service instance 1) on which MAC security is configured, the same MAC address cannot be learned on another service instance (for example, service instance 2) in the same bridge domain.

If service instance 2 attempts to learn the same MAC address, the violation response configured on service instance 2 is triggered. If MAC security is not configured on service instance 2 and a violation response is not configured, the “shutdown” response sequence is triggered on service instance 2.

If MAC security is not enabled on service instance 1, the violation is not triggered. service instance 2 learns the MAC address and moves it from service instance 1.

For some platforms, MAC address moves are allowed but moves between secured service instances and nonsecured service instances cannot be detected.

For example, if you do not configure MAC security on service instance 2 because of a hardware limitation, a MAC move from secured service instance 1 to service instance 2 is accepted. Therefore, it is recommended that all service instances within the same bridge-domain be configured as secured service instances.

Violation Response Configuration

A violation response is a response to a MAC security violation or a failed attempt to dynamically learn a MAC address due to an address violation. MAC security violations are of two types:

Type 1 Violation --The address of the ingress frame cannot be dynamically learned due to a deny list, or because doing so would cause the maximum number of secure addresses to be exceeded .

Type 2 Violation --The address of the ingress frame cannot be dynamically learned because it is already “present” on another secured service instance .

There are three possible sets of actions that can be taken in response to a violation:

1. **Shutdown**
    - The ingress frame is dropped.
    - The service instance on which the offending frame arrived is shut down.
    - The event and the response are logged to SYSLOG.

2. **Restrict**
    - The ingress frame is dropped.
    - The event and the response are logged to SYSLOG.

3. **Protect**
    - The ingress frame is dropped.

Note The ingress frame is dropped silently, without sending any violation report to the SYSLOG.

If a violation response is not configured, the default response mode is shutdown. The violation response can be configured to protect or restrict mode. A “no” form of a violation response, sets the violation response to the default mode of shutdown.

You are allowed to configure the desired response for a Type 1 and Type 2 violations on a service instance. For a Type 1 violation on a bridge domain (that is, if the learn attempt conforms to the policy configured on the service instance, but violates the policy configured on the bridge domain), the response is always “Protect.” This is not configurable.

In Restrict mode, the violation report is sent to SYSLOG at level LOG_WARNING.

Support for the different types of violation responses depends on the capabilities of the platform. The desired violation response can be configured on the service instance. The configured violation response does not take effect unless and until MAC security is enabled using the **mac security** command.

MAC Address Aging Configuration

A specific time scheduler can be set to age out secured MAC addresses that are dynamically learned or statically configured on both service instances and bridge domains, thus freeing up unused addresses from the MAC address table for other active subscribers.

The set of rules applied to age out secured MAC addresses is called secure aging. By default, the entries in the MAC address table of a secured service instance are never aged out. This includes permitted addresses and dynamically learned addresses.

The **mac security aging time** *aging-time* command sets the aging time of the addresses in the MAC address table to <*n* > minutes. By default, this affects only dynamically learned (not including sticky) addresses--permitted addresses and sticky addresses are not affected by the application of this command.

By default, the aging time <n> configured via the **mac security aging time** *aging-time* command is an absolute time. That is, the age of the MAC address is measured from the instant that it was first encountered on the service instance. This interpretation can be modified by using the **mac security aging time** *aging-time* **inactivity** command, which specifies that the age <n> be measured from the instant that the MAC address was last encountered on the service instance.

The **mac security aging static**and **mac security aging sticky** commands specify that the **mac security aging time**aging-time command must be applicable to permitted and sticky MAC addresses, respectively. In the case of permitted MAC addresses, the absolute aging time is measured from the time the address is entered into the MAC address table (for example, when it is configured or whenever the **mac security** command is entered--whichever is later).

If the **mac security aging time** command is not configured, the **mac security aging static** command has no effect.

Sticky MAC Address Configurations

The ability to make dynamically learned MAC addresses on secured service instances permanent even after interface transitions or device reloads can be set up and configured. A dynamically learned MAC address that is made permanent on a secured service instance is called a “sticky MAC address”. The **mac security sticky** command is used to enable the sticky MAC addressing feature on a service instance.

With the “sticky” feature enabled on a secured service instance, MAC addresses learned dynamically on the service instance are kept persistent across service instance line transitions and device reloads.

The sticky feature has no effect on statically configured MAC addresses. The sticky addresses are saved in the running configuration. Before the device is reloaded, it is the responsibility of the user to save the running configuration to the startup configuration. Doing this will ensure that when the device comes on, all the MAC addresses learned dynamically previously are immediately populated into the MAC address table.

The **mac security sticky address** *mac-address* command can configure a specific MAC address as a sticky MAC address. The use of this command is not recommended for the user because configuring a MAC address as a static address does the same thing. When sticky MAC addressing is enabled by the **mac security sticky** command, the dynamically learned addresses are marked as sticky and a **mac security sticky address** *mac-address* command is automatically generated and saved in the running configuration for each learned MAC address on the service instances.

Aging for Sticky Addresses

MAC addresses learned on a service instance that has the sticky behavior enabled are subject to aging as configured by the **mac security aging time** and **mac security aging sticky** commands. In other words, for the purpose of aging functionality, sticky addresses are treated the same as dynamically learned addresses.

Transitions

This section contains a description of the expected behavior of the different MAC security elements when various triggers are applied; for example, configuration changes or link state transitions.

MAC Security Enabled on a Service Instance

When MAC security is enabled on a service instance, all existing MAC table entries for the service instance are purged. Then, permitted MAC address entries and sticky addresses are added to the MAC table, subject to the prevailing MAC address limiting constraints on the bridge domain.

If MAC address limits are exceeded, any MAC address that fails to get added is reported via an error message to the console, the attempt to enable MAC security on the service instance fails, and the already added permitted entries are backed out or removed.

The aging timer for all entries is updated according to the secure aging rules.

MAC Security Disabled on a Service Instance

The existing MAC address table entries for this service instance are purged.

Service Instance Moved to a New Bridge Domain

This transition sequence applies to all service instances, whether or not they have MAC security configured. All the MAC addresses on this service instance in the MAC address table of the old bridge domain are removed. The count of dynamically learned addresses in the old bridge domain is decremented. Then, all the MAC security commands are permanently erased from the service instance.

Service Instance Removed from a Bridge Domain

All the MAC addresses in the MAC address table that attributable to this service instance are removed, and the count of dynamically learned addresses in the bridge domain is decremented. Since MAC security is applicable only on service instances that are members of a bridge domain, removing a service instance from a bridge domain causes all the MAC security commands to be erased permanently.

Service Instance Shut Down Due to Violation

All dynamically learned MAC addresses in the MAC address table are removed, and all the other MAC security state values are left unchanged. The only change is that no traffic is forwarded, and therefore no learning can take place.

Interface Service Instance Down Linecard OIR Removed

The MAC tables of all the affected bridge domains are cleared of all the entries attributable to the service instances that are down.

Interface Service Instance Re-activated Linecard OIR Inserted

The static and sticky address entries in the MAC tables of the affected bridge domains are re-created to the service instances that are activated.

MAC Address Limit Decreased

When the value of the MAC address limit on the service instance is changed initially, a sanity check is performed to ensure that the new value of <n> is greater than or equal to the number of permitted entries. If not, the command is rejected. The MAC table is scanned for addresses that are attributable to this service instance, and dynamically learned MAC addresses are removed when the new MAC address limit is less than the old MAC address limit.

Sticky Addresses Added or Removed on a Service Instance

Existing dynamically learned MAC addresses remain unchanged. All new addresses learned become “sticky” addresses.

Disabling sticky addresses causes all sticky secure MAC addresses on the service instance to be removed from the MAC address table. All new addresses learned become dynamic addresses on the service instance and are subject to aging.

How to Configure MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

Enabling MAC Security on a Service Instance

Perform this task to enable MAC address security on a service instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*

7. **mac security**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 8	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Enabling MAC Security on an EVC Port Channel

Before you begin

Note

- Bridge-domain, xconnect, and Ethernet virtual circuits (EVCs) are allowed only over the port channel interface and the main interface.
- If you configure a physical port as part of a channel group, you cannot configure EVCs under that physical port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *channel-group*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security**
8. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-group* **Example:** `Device(config)# interface port-channel 2`	Specifies the port channel group number and enters interface configuration mode. • Acceptable values are integers from 1 to 64.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance on an interface and enters service instance configuration mode.

	Command or Action	Purpose
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 8	end
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuring a MAC Address Permit List

Perform this task to configure permitted MAC addresses on a service instance that is a member of a bridge domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security address permit** *mac-address*
8. **mac security address permit** *mac-address*
9. **mac security address permit** *mac-address*
10. **mac security address permit** *mac-address*
11. **mac security address permit** *mac-address*
12. **mac security**
13. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used for mapping ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaaa`	Adds the specified MAC address as a permit MAC address for the service instance.
Step 8	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaab`	Adds the specified MAC address as a permitted MAC address for the service instance.
Step 9	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaac`	Adds the specified MAC address as a permitted MAC address for the service instance.

	Command or Action	Purpose
Step 10	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaad`	Adds the specified MAC address as a permitted MAC address for the service instance.
Step 11	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaae`	Adds the specified MAC address as a permitted MAC address for the service instance.
Step 12	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 13	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuring a MAC Address Deny List

Perform this task to configure a list of MAC addresses that are not allowed on a service instance that is a member of a bridge domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security address deny** *mac-address*
8. **mac security address deny** *mac-address*
9. **mac security address deny** *mac-address*
10. **mac security address deny** *mac-address*
11. **mac security address deny** *mac-address*
12. **mac security**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaaa`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 8	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaab`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 9	**mac security address deny** *mac-address* **Example:**	Adds the specified MAC address as a denied MAC address for the service instance.

	Command or Action	Purpose
	`Device(config-if-srv)# mac security address deny a2aa.aaaa.aaac`	
Step 10	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaad`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 11	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaae`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 12	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 13	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuring MAC Address Limiting on a Bridge Domain

Perform this task to configure an upper limit for the number of secured MAC addresses that reside in a bridge domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bridge-domain** *bridge-id*
4. **mac limit maximum addresses** *maximum-addresses*
5. end

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**bridge-domain** *bridge-id* **Example:** `Device(config)# bridge-domain 100`	Configures components on a bridge domain and enters bridge-domain configuration mode.
Step 4	**mac limit maximum addresses** *maximum-addresses* **Example:** `Device(config-bdomain)# mac limit maximum addresses 200`	Sets the MAC limit maximum addresses.

Step 5	end
	Example:
	`Device(config-bdomain)# end`
		Returns to user EXEC mode.

Configuring MAC Address Limiting on a Service Instance

Perform this task to configure an upper limit for the number of secured MAC addresses allowed on a service instance. This number includes addresses added as part of a permit list as well as dynamically learned MAC addresses. If the upper limit is decreased, all learned MAC entries are removed. If the upper limit is decreased, one or more learned MAC entries may be removed. The EFP secure MAC address limitation range is [1-1024], so the maximum is 1024.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security maximum addresses**
8. **mac security**
9. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security maximum addresses** **Example:** `Device(config-if-srv)# mac security maximum addresses 500`	Sets the maximum number of secure addresses permitted on the service instance.
Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 9	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuring a MAC Address Violation

Perform this task to specify the expected behavior of a device when an attempt to dynamically learn a MAC address fails because the configured MAC security policy on the service instance was violated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. Do one of the following:
    - **mac security violation restrict**
    - **mac security violation protect**
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.

	Command or Action	Purpose
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.

Step 7	Do one of the following:
	• **mac security violation restrict**
	• **mac security violation protect**
	Example:
	`Device(config-if-srv)# mac security violation restrict`
	Example:
	`Device(config-if-srv)# mac security violation protect`
		Sets the violation mode (for Type 1 and 2 violations) to restrict.
		or
		Sets the violation mode (for Type 1 and 2 violations) to protect.
		• If a MAC security violation response is not specified, by default, the violation mode is shutdown.

Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 9	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuring MAC Address Aging

Perform this task to configure the aging of secured MAC addresses under MAC security. Secured MAC addresses are not subject to the normal aging of MAC table entries. If aging is not configured, secured MAC addresses are never aged out.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security aging time** *aging-time* **[inactivity]**
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security aging time** *aging-time* **[inactivity]** **Example:** `Device(config-if-srv)# mac security aging time 200 inactivity`	Sets the aging time for secure addresses, in minutes. The optional **inactivity** keyword specifies that the aging out of addresses is based on inactivity of the sending hosts (as opposed to absolute aging).
Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 9	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuring a Sticky MAC Address

If sticky MAC addressing is configured on a secured service instance, MAC addresses that are learned dynamically on the service instance are retained during a link-down condition. Perform this task to configure sticky MAC addresses on a service instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security sticky address** *mac-address*
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.

	Command or Action	Purpose
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security sticky address** *mac-address* **Example:** `Device(config-if-srv)# mac security sticky address 1111.2222.3333`	Sets up a MAC address to be declared as a sticky MAC address on the service instance.
Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 9	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Displaying the MAC Security Status of a Specific Service Instance

Perform this task to display the MAC security status of a service instance.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type* *number* **mac security**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type* *number* **mac security** **Example:** `Device# show ethernet service instance id 100 interface gigabitethernet1/1 mac security`	Displays the MAC security status of a specific service instance.

	Command or Action	Purpose

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Displaying the Service Instances with MAC Security Enabled

Perform this task to display all the service instances with MAC security enabled.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance mac security**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance mac security** **Example:** `Device# show ethernet service instance mac security`	Displays all the service instances with MAC security enabled.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Displaying the Service Instances with MAC Security Enabled on a Specific Bridge Domain

Perform this task to display the service instances on a specific bridge domain that have MAC security enabled.

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *id* **mac security**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bridge-domain** *id* **mac security** **Example:** `Device# show bridge-domain 100 mac security`	Displays all the service instances with MAC security enabled on a specific bridge domain.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Showing the MAC Addresses of All Secured Service Instances

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance mac security address**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance mac security address** **Example:** `Device# show ethernet service instance mac security address`	Displays the secured addresses on all the service instances.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Showing the MAC Addresses of a Specific Service Instance

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type* *number* **mac security address**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type* *number* **mac security address** **Example:** `Device# show ethernet service instance id 200 interface GigabitEthernet 1/0 mac security address`	Displays the addresses of a specific service instance.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Showing the MAC Addresses of All Service Instances on a Specific Bridge Domain

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *id* **mac security address**
3. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**show bridge-domain** *id* **mac security address** **Example:** `Device# show bridge-domain 100 mac security address`	Displays the secured addresses of all the service instances on a specified bridge domain.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Showing the MAC Security Statistics of a Specific Service Instance

This section describes how to display the MAC security statistics of a specific service instance.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type number* **mac security statistics**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type number* **mac security statistics** **Example:** `Device# show ethernet service instance id 100 interface gigabitethernet1/1 mac security statistics`	Displays the MAC security statistics of a specific service instance.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Showing the MAC Security Statistics of All Service Instances on a Specific Bridge Domain

Perform this task to display the MAC security statistics of all the service instances on a specific bridge domain.

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *bridge-id* **mac security statistics**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bridge-domain** *bridge-id* **mac security statistics** **Example:** `Device# show bridge-domain 100 mac security statistics`	Displays the MAC security statistics of all service instances that belong to a specific bridge domain.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Showing the Last Violation Recorded on Each Service Instance on a Specific Bridge Domain

Perform this task to display the last violation recorded on each service instance on a specific bridge domain. Service instances on which there have been no violations are excluded from the output.

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *bridge-id* **mac security last violation**
3. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**show bridge-domain** *bridge-id* **mac security last violation** **Example:** `Device# show bridge-domain 100 mac security last violation`	Displays information about the last violation recorded on each of the service instances that belong to the bridge domain.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Clearing All Dynamically Learned Secure MAC Addresses on a Service Instance

Perform this task to clear all dynamically learned Secure MAC addresses on a service instance.

SUMMARY STEPS

1. **enable**
2. **clear ethernet service instance id** *id* **interface** *type number* **mac table**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ethernet service instance id** *id* **interface** *type number* **mac table** **Example:** `Device# clear ethernet service instance id 100 interface gigabitethernet0/0/1 mac table`	Clears all the dynamically learned Secure MAC addresses on the specified service instance.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Clearing All Dynamically Learned MAC Addresses on a Bridge Domain

Perform this task to clear all dynamically learned MAC addresses on a bridge domain.

SUMMARY STEPS

1. **enable**
2. **clear bridge-domain** *bridge-id* **mac table**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bridge-domain** *bridge-id* **mac table** **Example:** `Device# clear bridge-domain 100 mac table`	Clears all dynamically learned MAC addresses on the specified bridge domain.

Step 3	**end**
	Example:
	`Device# end`
		Returns to user EXEC mode.

Bringing a Specific Service Instance Out of the Error-Disabled State

Perform this task to bring a specific service instance out of the error-disabled state.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security**
8. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge-domain instance where *bridge-id* is the identifier for the bridge-domain instance.
Step 7	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.

Step 8	**end**
	Example:
	`Device(config-if-srv)# end`
		Returns to user EXEC mode.

Configuration Examples for MAC Address Limiting on Service Instances and Bridge Domains and EVC Port Channels

Example Enabling MAC Security on a Service Instance

The following example shows how to enable MAC security on a service instance:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Enabling MAC Security on an EVC Port Channel

The following example shows how to enable MAC Security on an EVC port channel:

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a MAC Address Permit List

The following example shows how to configure a MAC address permit list:

Example Configuring a MAC Address Deny List

The following example shows how to configure a MAC address deny list:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaaa
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaab
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaac
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaad
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaae
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring MAC Address Limiting on a Bridge Domain

```
Device> enable
Device# configure terminal
Device(config)# bridge-domain 100
Device(config-bdomain)# mac limit maximum addresses 1000
Device(config-bdomain)# end
```

Example Configuring a MAC Address Limit on a Service Instance

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security maximum addresses 10
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a MAC Address Violation Response

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaaa
Device(config-if-srv)# mac security violation protect
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring MAC Address Aging

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 4/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security aging time 10
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a Sticky MAC Address

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security sticky address 1111.2222.3333
Device(config-if-srv)# mac security
```

Additional References

Related Documents

Related Topic	Document Title
CFM commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*
Configuring Ethernet connectivity fault management in a service provider network (Cisco pre-Standard CFM Draft 1)	"Configuring Ethernet Connectivity Fault Management in a Service Provider Network” module in the *Cisco IOS Carrier Ethernet Configuration Guide*
Ethernet Local Management Interface on a provider edge device	"Configuring Ethernet Local Management Interface on a Provider Edge Device” module in the *Cisco IOS Carrier Ethernet Configuration Guide*
IP SLAs for Metro Ethernet	"IP SLAs for Metro Ethernet"
NSF/SSO and MPLS	"NSF/SSO - MPLS LDP and LDP Graceful Restart"
ISSU feature and functions	"Cisco IOS Broadband High Availability In Service Software Upgrade"
Performing an ISSU	"Cisco IOS In Service Software Upgrade Process and Enhanced Fast Software Upgrade Process"
SSO	“Stateful Switchover” chapter of the *Cisco IOS High Availability Configuration Guide*

Standards

Standard	Title
IEEE 802.1ag Standard	*802.1ag - Connectivity Fault Management*
IEEE 802.3ah	*IEEE 802.3ah Ethernet in the First Mile*
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	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Technical Assistance

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

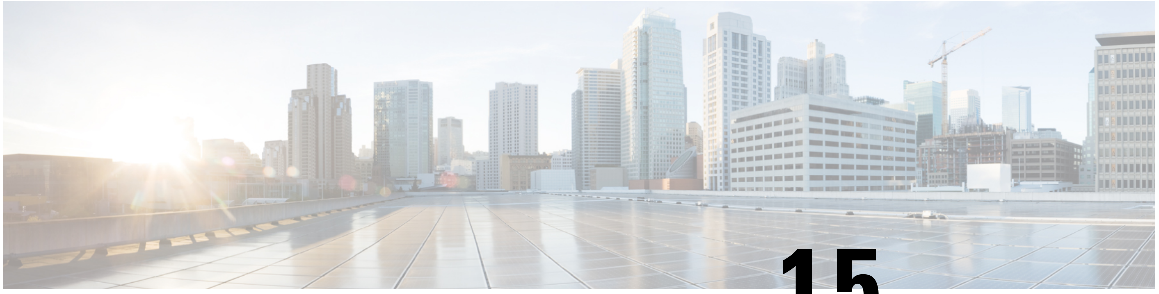
Feature Information for MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

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

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

Table 23: Feature Information for MAC Address Limiting on Service Instances, Bridge Domains, and EVC Port Channels

Feature Name	Releases	Feature Information
MAC Address Limiting on Service Instances and Bridge Domains	Cisco IOS XE 3.7S Cisco IOS XE Gibraltar 16.11.a	The MAC Address Limiting on Service Instances and Bridge Domains feature addresses port security with service instances by providing the capability to control and filter MAC address learning behavior at the granularity of a per-service instance. When a violation requires a shutdown, only the customer that is assigned to a given service instance is affected. MAC address limiting is a type of MAC security and is also referred to as a MAC security component or element. The following commands were introduced or modified: **bridge-domain (global)**, **bridge-domain (service instance)**, **clear bridge-domain mac-table**, **clear ethernet service instance**, **errdisable recovery cause mac-security**, **interface**, **mac limit maximum addresses**,**mac security**,**show bridge-domain**, **show ethernet service instance**. Support was added in Cisco IOS XE Gibraltar 16.11.a for Cisco ASR 1000 Series Aggregation Services Routers, Cisco Cloud Services Router 1000v, and Cisco 4000 Series Integrated Services Routers.



CHAPTER 15

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.

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.

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.

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** *csi-id* **evc** *evc-name* **vlan** *vlan-id*
5. **continuity-check**
6. **continuity-check interval** *time*
7. **exit**

8. **exit**
9. **ethernet evc** *evc-id*
10. **oam protocol** {**cfm domain** *domain-name* | **ldp**}
11. **uni count** *value* [**multipoint**]
12. **exit**
13. Repeat Steps 3 through 12 to define other CFM domains that you want OAM manager to monitor.
14. **interface** *type number*
15. **service instance** *id* **ethernet** [*evc-id*]
16. **ethernet lmi ce-vlan map** {*vlan-id* [**untagged**] | **any** | **default** | **untagged**}
17. **ethernet lmi interface**
18. **encapsulation dot1q** *vlan-id*
19. **bridge-domain** *domain-number*
20. **cfm mep domain** *domain-name* **mpid** *mpid-id*
21. **exit**
22. **service instance** *service-instance-id* **ethernet**
23. **encapsulation untagged**
24. **l2protocol peer**
25. **bridge-domain** *bridge-domain-number*
26. **exit**
27. **ethernet uni** [**bundle** [**all-to-one**] | **id** *uni-id* | **multiplex**]
28. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.

	Command or Action	Purpose
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* [**multipoint**] **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.

	Command or Action	Purpose
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:**	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	**ethernet lmi interface** **Example:** `Device(config-if-srv)# ethernet lmi interface`	Enables Ethernet local management interface (LMI) on a UNI.
Step 18	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 2`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance.
Step 19	**bridge-domain** *domain-number* **Example:** `Device(config-if-srv)# bridge-domain 1`	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
Step 20	**cfm mep domain** *domain-name* **mpid** *mpid-id* **Example:** `Device(config-if-srv)# cfm mep domain provider mpid 10`	Configures a maintenance endpoint (MEP) for a domain.
Step 21	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.
Step 22	**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 23	**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 24	**l2protocol peer** **Example:** `Device(config-if-srv)# l2protocol peer`	Configures transparent Layer 2 protocol peering on the interface.
Step 25	**bridge-domain** *bridge-domain-number* **Example:** `Device(config-if-srv)# bridge-domain 1`	Binds a service instance to a bridge domain instance.
Step 26	**exit** **Example:** `Device(config-if)# exit`	Returns to interface configuration mode.
Step 27	**ethernet uni** [**bundle** [**all-to-one**] | **id** *uni-id* | **multiplex**] **Example:** `Device(config-if)# ethernet uni bundle`	Sets UNI bundling attributes.

Step 28	**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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet lmi interface**
5. **ethernet lmi** {**n393** *value* | **t392** *value*}
6. **end**

DETAILED STEPS

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

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

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.

SUMMARY STEPS

1. **enable**
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*]}
3. **show ethernet service evc** [**detail** \| **id** *evc-id* [**detail**] \| **interface** *type number* [**detail**]]
4. **show ethernet service instance** [**detail** \| **id** *id* \| **interface** *type number* \| **policy-map** \| **stats**]
5. **show ethernet service interface** [*type number*] [**detail**]

DETAILED STEPS

	Command or Action	Purpose
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.

	Command or Action	Purpose
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#

Interface: Gigabitethernet
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: GigabitEthernet
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)#
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
```

This example shows a configuration of OAM manager, CFM, and Ethernet LMI over an Xconnect configuration:

```
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
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)# oam protocol cfm domain provider
Device(config-evc)# exit
Device(config)#
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 ethernet
Device(config-if-srv)# ethernet lmi ce-vlan map 101
Device(config-if-srv)# encapsulation dot1q 2
Device(config-if-srv)# xconnect 10.1.1.1 100 encapsulation mpls
Device(cfg-if-ether-vc-xconn)# exit
Device(config-if-srv)# cfm mep domain provider mpid 10
Device(config-if-srv-ecfm-mep)# end
```

Example: Ethernet LMI on a CE Device Configuration

This example shows how to configure Ethernet Local Management Interface (LMI) globally on a customer edge (CE) device:

```
Device# configure terminal
Device(config)# ethernet lmi global
Device(config)# ethernet lmi ce
Device(config)# exit
```

Additional References for Configuring Ethernet Local Management Interface at a Provider Edge

Related Documents

Related Topic	Document Title
Ethernet Connectivity Fault Management (CFM)	"Configuring Ethernet Connectivity Fault Management in a Service Provider Network" in the *Carrier Ethernet Configuration Guide*
Ethernet Local Management Interface (LMI)	"Enabling Ethernet Local Management Interface" in the *Carrier Ethernet Configuration Guide*
Remote Port Shutdown feature	"Configuring Remote Port Shutdown" in the *Carrier Ethernet Configuration Guide*
IEEE 802.3ah	*IEEE 802.3ah Ethernet in the First Mile*
Cisco high availability (HA) configuration information	*High Availability Configuration Guide*
Ethernet LMI 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/D5.2	*Draft Standard for Local and Metropolitan Area Networks*
ITU-T	*ITU-T Y.1731 OAM Mechanisms for Ethernet-Based Networks*
IETF VPLS OAM	*L2VPN OAM Requirements and Framework*
Metro Ethernet Forum 16 Technical Specification	*Technical Specification MEF 16- Ethernet Local Management Interface*
ITU-T Q.3/13	*Liaison statement on Ethernet OAM (Y.17ethoam)*

Technical Assistance

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

Feature Information for Configuring Ethernet Local Management Interface at a Provider Edge

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

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

Table 24: Feature Information for Configuring Ethernet Local Management Interface at a Provider Edge

Feature Name	Releases	Feature Information
Ethernet Local Management Interface at a Provider Edge	12.2(33)SRB 12.2(33)SXI	Ethernet LMI is an Ethernet OAM protocol between a CE device and a PE device. Ethernet LMI provides CE devices with the status of EVCs for large Ethernet MANs and WANs and provides information that enables CE devices to autoconfigure. Specifically, Ethernet LMI runs on the PE-CE UNI link and notifies a CE device of the operating state of an EVC and when an EVC is added or deleted. Ethernet LMI also communicates the attributes of an EVC. In Cisco IOS Release 12.2(33)SRB, this feature was introduced on the Cisco 7600 series router. The following commands were introduced or modified: **debug ethernet lmi**, **debug ethernet service, ethernet evc**, **ethernet lmi ce-vlan map**, **ethernet uni**, **oam protocol**, **service instance ethernet**, **show ethernet service evc**, **show ethernet service instance**, **show ethernet service interface, uni count**.
ISSU Support in E-LMI	12.2(33)SRD 15.0(1)S	ISSU allows you to perform a Cisco IOS software upgrade or downgrade without disrupting packet flow. ISSU lowers the impact that planned maintenance activities have on network availability by allowing software changes while the system is in service. In Cisco IOS Release 12.2(33)SRD, this feature was introduced on the Cisco 7600 series router. The following commands were introduced or modified: **debug ethernet lmi**.
NSF/SSO Support in E-LMI	12.2(33)SRD 15.0(1)S	The redundancy configurations SSO and NSF are supported in Ethernet LMI and are automatically enabled. A switchover from an active to a standby RP occurs when the active RP 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 switchover. NSF also interoperates with the SSO feature to minimize network downtime following a switchover. In Cisco IOS Release 12.2(33)SRD, this feature was introduced on the Cisco 7600 series router. The following commands were introduced or modified: **debug ethernet lmi**.

CHAPTER 16

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.

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.

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

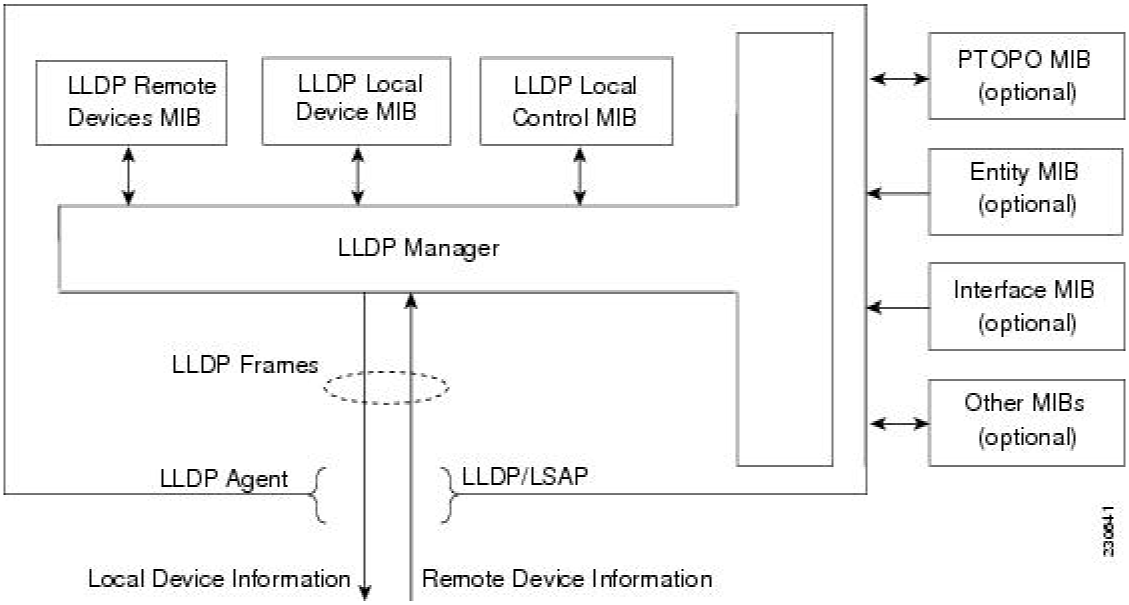
IEEE 802.1ab LLDP

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

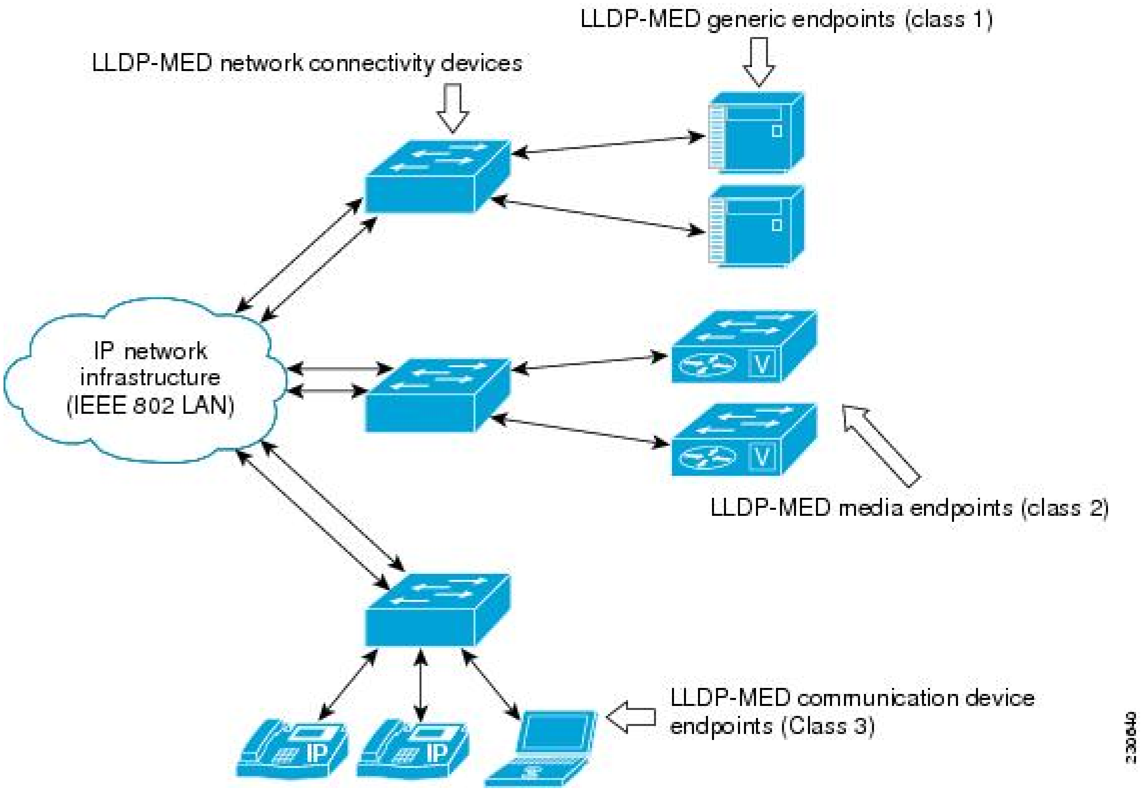
LLDP-MED operates between several classes of network equipment such as IP phones, conference bridges, and network connectivity devices such as routers and switches. By default, a network connectivity device sends out only LLDP packets until it receives LLDP-MED packets from an endpoint device. The network 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.


LLDP-MED generic endpoints (class 1)
LLDP-MED network connectivity devices
IP network infrastructure (IEEE 802 LAN)
LLDP-MED media endpoints (class 2)
LLDP-MED communication device endpoints (Class 3)
IP
IP
230640

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **lldp run**

4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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 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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no lldp** {**med-tlv-select** *tlv* | **receive** | **transmit**}
5. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **lldp holdtime** *seconds*
4. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device# configure terminal`	
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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **lldp timer** *rate*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**lldp timer** *rate* **Example:** `Device(config)# lldp timer 75`	Specifies the rate at which LLDP packets are sent every second.
Step 4	**end** **Example:**	Returns to privileged EXEC mode.

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

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.

SUMMARY STEPS

1. **enable**
2. **show lldp** [**entry** {***** | *word*} | **errors** | **interface** [**ethernet** *number*]| **neighbors** [**ethernet** *number*| **detail**]| **traffic**]
3. **clear lldp** {**counters** | **table**}
4. **end**

DETAILED STEPS

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **lldp tlv-select** *tlv*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **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.

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **lldp med-tlv-select** *tlv*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **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.

Configuration Examples for Link Layer Discovery Protocol in Multivendor Networks

Example: Configuring Voice VLAN

The following example shows how to configure voice VLAN and verify

```
Device1> enable
Device1# configure terminal
```

```
Device1(config)# interface GigabitEthernet0/1/7
Device1(config-if)# switchport voice vlan 10
Device1(config-if)# no ip address
Device1(config-if)# end
```

The following example displays the updated running configuration on Device 2. LLDP is enabled with hold time, timer, and TLV options configured.

```
Device1# show lldp neighbors detail

Local Intf: Gi0/1/7
Chassis id: 10.10.0.1
Port id: C8F9F9D61BC2:P1
Port Description: SW PORT
System Name: SEPC8F9F9D61BC2

System Description:
Cisco IP Phone 7962G,V12, SCCP42.9-3-1ES27S

Time remaining: 127 seconds
System Capabilities: B,T
Enabled Capabilities: B,T
Management Addresses:
    IP: 10.10.0.1
Auto Negotiation - supported, enabled
Physical media capabilities:
    1000baseT(HD)
    1000baseX(FD)
    Symm, Asym Pause(FD)
    Symm Pause(FD)
Media Attachment Unit type: 16
Vlan ID: - not advertised

MED Information:

    MED Codes:
          (NP) Network Policy, (LI) Location Identification
          (PS) Power Source Entity, (PD) Power Device
          (IN) Inventory

    H/W revision: 12
    F/W revision: tnp62.8-3-1-21a.bin
    S/W revision: SCCP42.9-3-1ES27S
    Serial number: FCH1610A5S5
    Manufacturer: Cisco Systems, Inc.
    Model: CP-7962G
    Capabilities: NP, PD, IN
    Device type: Endpoint Class III
    Network Policy(Voice): VLAN 10, tagged, Layer-2 priority: 5, DSCP: 46
    Network Policy(Voice Signal): VLAN 10, tagged, Layer-2 priority: 4, DSCP: 32
    PD device, Power source: Unknown, Power Priority: Unknown, Wattage: 6.3
    Location - not advertised
```

The following example shows how to configure LLDP timer, hold time, and TLVs options on Device 2.

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# lldp run
Device(config)# lldp holdtime 150
Device(config)# lldp timer 15
Device(config)# lldp tlv-select port-vlan
Device(config)# lldp tlv-select mac-phy-cfg
Device2(config)# interface ethernet 0/0
```

```
Device2(config-if)# lldp transmit
Device2(config-if)# end
00:08:32: %SYS-5-CONFIG_I: Configured from console by console
```

The following example shows that voice vlan has been configured on the IP phone.

```
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
```

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
```

Feature Information for Link Layer Discovery Protocol in Multivendor Networks

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

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

Table 25: Feature Information for Using Link Layer Discovery Protocol in Multivendor Networks

Feature Name	Releases	Feature Information
IEEE 802.1ab LLDP (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 SNMP in multivendor networks. The following commands were introduced or modified: **clear lldp**, **lldp** and **show lldp**.

Feature Name	Releases	Feature Information
ANSI TIA-1057 LLDP-MED Support		MED is an LLDP enhancement that was formalized by the TIA for VoIP applications. The Cisco implementation of LLDP is based on the IEEE 802.1ab standard. The following commands were introduced or modified: **lldp** and **lldp** (interface).
IEEE 802.1ab LLDP (Link Layer Discovery Protocol)	Cisco IOS XE Release 3.2E Cisco IOS XE Release 3.6E	IEEE 802.3ad link bundling and load balancing leverages the EtherChannel infrastructure within Cisco software to manage the bundling of various links. The network traffic load-balancing features help minimize network disruption that results when a port is added or deleted from a link bundle. MED is an LLDP enhancement that was formalized by the TIA for VoIP applications. In Cisco IOS XE Release 3.2SE, this feature is supported on the following platforms: • Cisco 5700 Series Wireless LAN Controllers • Cisco Catalyst 3850 Series Switches In Cisco IOS XE Release 3.3SE, this feature is supported on the following platforms: • Cisco Catalyst 3650 Series Switches
LLDP MED Support on ISRG2		The LLDP MED feature is supported on Cisco Integrated Services Routers Generation 2 (ISR G2). No commands were introduced or modified.

CHAPTER 17

Multichassis LACP

In Carrier Ethernet networks, various redundancy mechanisms provide resilient interconnection of nodes and networks. The choice of redundancy mechanisms depends on various factors such as transport technology, topology, single node versus entire network multihoming, capability of devices, autonomous system (AS) boundaries or service provider operations model, and service provider preferences.

Carrier Ethernet network high-availability can be achieved by employing both intra- and interchassis redundancy mechanisms. Cisco's Multichassis EtherChannel (MCEC) solution addresses the need for interchassis redundancy mechanisms, where a carrier wants to "dual home" a device to two upstream points of attachment (PoAs) for redundancy. Some carriers either cannot or will not run loop prevention control protocols in their access networks, making an alternative redundancy scheme necessary. MCEC addresses this issue with enhancements to the 802.3ad Link Aggregation Control Protocol (LACP) implementation. These enhancements are provided in the Multichassis LACP (mLACP) feature described in this document.

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.

Prerequisites for mLACP

- The command **lacp max-bundle** must be used on all PoAs in order to operate in PoA control and shared control modes.
    - The maximum number of links configured cannot be less than the total number of interfaces in the link aggregation group (LAG) that is connected to the PoA.
    - Each PoA may be connected to a dual-homed device (DHD) with a different number of links for the LAG (configured with a different number of maximum links).
- Each PoA must be configured using the **lacp min-bundle** command with the desired minimum number of links to maintain the LAG in the active state.
- For DHD control there must be an equal number of links going to each PoA.
- The max-bundle value must equal the number of active links connected locally to the PoA (no local intra-PoA active or standby protection).
- LACP fast switchover must be configured on all devices to speed convergence.

Restrictions for mLACP

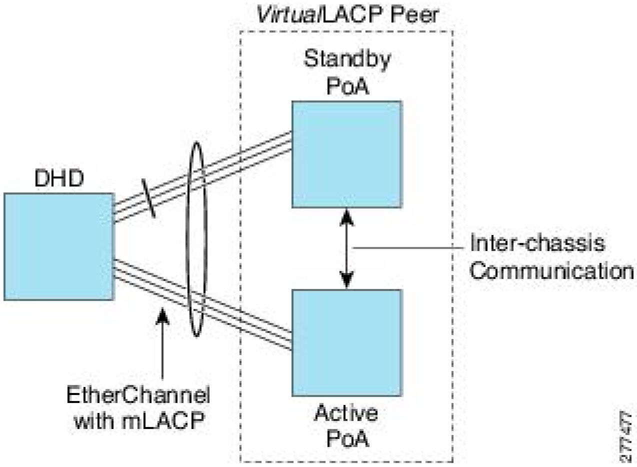
- mLACP does not support Fast Ethernet.
- mLACP does not support half-duplex links.
- mLACP does not support multiple neighbors.
- Converting a port channel to mLACP can cause a service disruption.
- The maximum number of member links per LAG per PoA is restricted by the maximum number of ports per port channel, as limited by the platform.
- System priority on a DHD must be a lesser priority than on PoAs.
- MAC Tunneling Protocol (MTP) supports only one member link in a port channel.
- A port-channel or its member links may flap while LACP stabilizes.
- DHD-based control does not function when min-links is not configured.
- DHD-controlled revertive behavior with min-links is not supported.
- Brute-force failover always causes min-link failures.
- Any failure with brute-force failover behaves revertively.

Information About mLACP

Overview of Multichassis EtherChannel

In Multichassis EtherChannel (MCEC), the DHD is dual-homed to two upstream PoAs. The DHD is incapable of running any loop prevention control protocol such as Multiple Spanning Tree (MST). Therefore, another mechanism is required to prevent forwarding loops over the redundant setup. One method is to place the DHD’s uplinks in a LAG, commonly referred to as EtherChannel. This method assumes that the DHD is capable of running only IEEE 802.3ad LACP for establishing and maintaining the LAG.

LACP, as defined in IEEE 802.3ad, is a link-level control protocol that allows the dynamic negotiation and establishment of LAGs. An extension of the LACP implementation to PoAs is required to convey to a DHD that it is connected to a single virtual LACP peer and not to two disjointed devices. This extension is called Multichassis LACP or mLACP. The figure below shows this setup.



The PoAs forming a virtual LACP peer, from the perspective of the DHD, are defined as members of a redundancy group. For the PoAs in a redundancy group to appear as a single device to the DHD, the states between them must be synchronized through the Interchassis Communication Protocol (ICCP), which provides a control-only interchassis communication channel (ICC).

In Cisco IOS Release 12.2(33)SRE, the system functions in active/standby redundancy mode. In this mode DHD uplinks that connect to only a single PoA can be active at any time. The DHD recognizes one PoA as active and the other as standby but does not preclude a given PoA from being active for one DHD and standby for another. This capability allows two PoAs to perform load sharing for different services.

Interactions with the MPLS Pseudowire Redundancy Mechanism

The network setup shown in the figure above can be used to provide provider edge (PE) node redundancy for Virtual Private LAN Service (VPLS) and Virtual Private Wire Service (VPWS) deployments over Multiprotocol Label Switching (MPLS). In these deployments, the uplinks of the PoAs host the MPLS pseudowires that provide redundant connectivity over the core to remote PE nodes. Proper operation of the network requires interaction between the redundancy mechanisms employed on the attachment circuits (for example, mLACP) and those employed on the MPLS pseudowires. This interaction ensures the state (active or standby) is synchronized between the attachment circuits and pseudowires for a given PoA.

RFC 4447 introduced a mechanism to signal pseudowire status via the Link Distribution Protocol (LDP) and defined a set of status codes to report attachment circuit as well as pseudowire fault information. The Preferential Forwarding Status bit (*draft-ietf-pwe3-redundancy-bit*) definition proposes to extend these codes to include two bits for pseudowire redundancy applications:

- Preferential forwarding status: active or standby
- Request pseudowire switchover

The draft also proposes two modes of operation:

- Independent mode--The local PE decides on its pseudowire status independent of the remote PE.
- Primary and secondary modes--One of the PEs determines the state of the remote side through a handshake mechanism.

For the mLACP feature, operation is based on the independent mode. By running ICC between the PoAs, only the preferential forwarding status bit is required; the request pseudowire switchover bit is not used.

The local pseudowire status (active or standby) is determined independently by the PoAs in a redundancy group and then relayed to the remote PEs in the form of a notification. Similarly, the remote PEs perform their own selection of their pseudowire status and notify the PoAs on the other side of the core.

After this exchange of local states, the pseudowires used for traffic forwarding are those selected to be active independently on both local and remote ends.

The attachment circuit redundancy mechanism determines and controls the pseudowire redundancy mechanism. mLACP determines the status of the attachment circuit on a given PoA according to the configured LACP system and port priorities, and then the status of the pseudowires on a given PoA is synchronized with that of the local attachment circuits. This synchronization guarantees that the PoA with the active attachment circuits has its pseudowires active. Similarly, the PoA with the standby attachment circuits has its pseudowires in standby mode. By ensuring that the forwarding status of the attachment circuits is synchronized with that of the pseudowires, the need to forward data between PoA nodes within a redundancy group can be avoided. This synchronization saves platform bandwidth that would otherwise be wasted on inter-PoA data forwarding in case of failures.

Redundancy Mechanism Processes

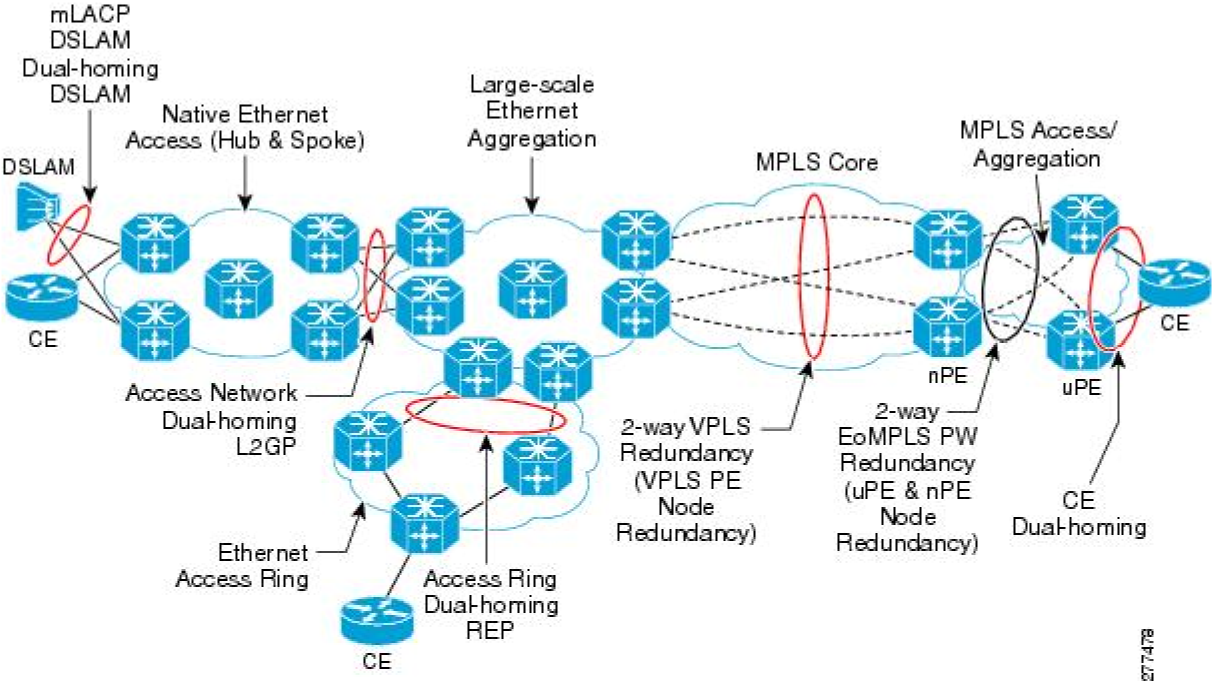
The Carrier Ethernet redundancy solution should include the following processes (and how they apply to the mLACP solution):

- Attachment circuit active or standby status selection--This selection can be performed by the access node or network, the aggregation node, or combination of the two. For mLACP, the attachment circuit status selection is determined through collaboration between the DHD and the PoAs.
- Pseudowire forwarding status notification--This notification is mandatory for mLACP operation in VPWS and VPLS deployments; that is, when the PoA uplinks employ pseudowire technology. When the PoAs decide on either an active or standby role, they need to signal the status of the associated pseudowires to the PEs on the far end of the network. For MPLS pseudowires, this is done using LDP.
- MAC flushing indication--This indication is mandatory for any redundancy mechanism in order to speed convergence time and eliminate potential traffic failure. The mLACP redundancy mechanism should be integrated with relevant 802.1Q/802.1ad/802.1ah MAC flushing mechanisms as well as MAC flushing mechanisms for VPLS.

Note Failure occurs when incoming traffic is dropped without informing the source that the data did not reach its intended recipient. Failure can be detected only when lost traffic is monitored.

- Active VLAN notification--For mLACP, this notification is not required as long as the PoAs follow the active/standby redundancy model.

The figure below shows redundancy mechanisms in Carrier Ethernet networks.



Dual-Homed Topology Using mLACP

The mLACP feature allows the LACP state machine and protocol to operate in a dual-homed topology. The mLACP feature decouples the existing LACP implementation from the multichassis specific requirements, allowing LACP to maintain its adherence to the IEEE 802.3ad standard. The mLACP feature exposes a single virtual instance of IEEE 802.3ad to the DHD for each redundancy group. The virtual LACP instance interoperates with the DHD according to the IEEE 802.3ad standard to form LAGs spanning two or more chassis.

LACP and 802.3ad Parameter Exchange

In IEEE 802.3ad, the concatenation of the LACP system MAC address and system priority form an LACP system ID (8 bytes). The system ID is formed by taking the two-byte system priority value as the most significant two octets of the system ID. The system MAC address makes up the remainder of the system ID (octets 3 to 8). System ID priority comparisons are based on the lower numerically valued ID.

To provide the highest LACP priority, the mLACP module communicates the system MAC address and priority values for the given redundancy group to its redundancy group peer(s) and vice versa. The mLACP then chooses the lowest system ID value among the PoAs in the given redundancy group to use as the system ID of the virtual LACP instance of the redundancy group.

Cisco IOS Release 12.2(33)SRE introduces two LACP configuration commands to specify the system MAC address and system priority used for a given redundancy group: **mlacp system-mac** *mac-address* and **mlacp system-priority** *priority-value*. These commands provide better settings to determine which side of the attachment circuit will control the selection logic of the LAG. The default value for the system MAC address is the chassis backplane default MAC address. The default value for the priority is 32768.

Port Identifier

IEEE 802.3ad uses a 4-byte port identifier to uniquely identify a port within a system. The port identifier is the concatenation of the port priority and port number (unique per system) and identifies each port in the system. Numerical comparisons between port IDs are performed by unsigned integer comparisons where the 2-byte Port Priority field is placed in the most significant two octets of the port ID. The 2-byte port number makes up the third and fourth octets. The mLACP feature coordinates the port IDs for a given redundancy group to ensure uniqueness.

Port Number

A port number serves as a unique identifier for a port within a device. The LACP port number for a port is equal to the port’s ifIndex value (or is based on the slot and subslot identifiers on the Cisco 7600 router).

LACP relies on port numbers to detect rewiring. For multichassis operation, you must enter the **mlacp node-id** *node-id* command to coordinate port numbers between the two PoAs in order to prevent overlap.

Port Priority

Port priority is used by the LACP selection logic to determine which ports should be activated and which should be left in standby mode when there are hardware or software limitations on the maximum number of links allowed in a LAG. For multichassis operation in active/standby redundancy mode, the port priorities for all links connecting to the active PoA must be higher than the port priorities for links connecting to the standby PoA. These port priorities can either be guaranteed through explicit configuration or the system can automatically adjust the port priorities depending on selection criteria. For example, select the PoA with the highest port priority to be the active PoA and dynamically adjust the priorities of all other links with the same port key to an equal value.

In Cisco IOS Release 12.2(33)SRE, the mLACP feature supports only the active/standby redundancy model. The LACP port priorities of the individual member links should be the same for each link belonging to the LAG of a given PoA. To support this requirement, the **mlacp lag-priority** command is implemented in interface configuration mode in the command-line interface (CLI). This command sets the LACP port priorities for all the local member links in the LAG. Individual member link LACP priorities (configured by the **lacp port-priority** command) are ignored on links belonging to mLACP port channels.

The **mlacp lag-priority** command may also be used to force a PoA failover during operation in the following two ways:

- Set the active PoA’s LAG priority to a value greater than the LAG priority on the standby PoA. This setting results in the quickest failover because it requires the fewest LACP link state transitions on the standby links before they turn active.
- Set the standby PoA’s LAG priority to a value numerically less than the LAG priority on the active PoA. This setting results in a slightly longer failover time because standby links have to signal OUT_OF_SYNC to the DHD before the links can be brought up and go active.

In some cases, the operational priority and the configured priority may differ when using dynamic port priority management to force failovers. In this case, the configured version will not be changed unless the port channel is operating in nonrevertive mode. Enter the **show lacp multichassis port-channel** command to view the current operational priorities. The configured priority values can be displayed by using the **show running-config** command.

Multichassis Considerations

Because LACP is a link layer protocol, all messages exchanged over a link contain information that is specific and local to that link. The exchanged information includes:

- System attributes--priority and MAC address
- Link attributes--port key, priority, port number, and state

When extending LACP to operate over a multichassis setup, synchronization of the protocol attributes and states between the two chassis is required.

System MAC Address

LACP relies on the system MAC address to determine the identity of the remote device connected over a particular link. Therefore, to mask the DHD from its connection to two disjointed devices, coordination of the system MAC address between the two PoAs is essential. In Cisco IOS software, the LACP system MAC address defaults to the ROM backplane base MAC address and cannot be changed by configuration. For multichassis operation the following two conditions are required:

- System MAC address for each PoA should be communicated to its peer--For example, the PoAs elect the MAC address with the lower numeric value to be the system MAC address. The arbitration scheme must resolve to the same value. Choosing the lower numeric MAC address has the advantage of providing higher system priority.
- System MAC address is configurable--The system priority depends, in part, on the MAC address, and a service provider would want to guarantee that the PoAs have higher priority than the DHD (for example, if both DHD and PoA are configured with the same system priority and the service provider has no control over DHD). A higher priority guarantees that the PoA port priorities take precedence over the DHD’s port priority configuration. If you configure the system MAC address, you must ensure that the addresses are uniform on both PoAs; otherwise, the system will automatically arbitrate the discrepancy, as when a default MAC address is selected.

System Priority

LACP requires that a system priority be associated with every device to determine which peer’s port priorities should be used by the selection logic when establishing a LAG. In Cisco IOS software, this parameter is configurable through the CLI. For multichassis operation, this parameter is coordinated by the PoAs so that the same value is advertised to the DHD.

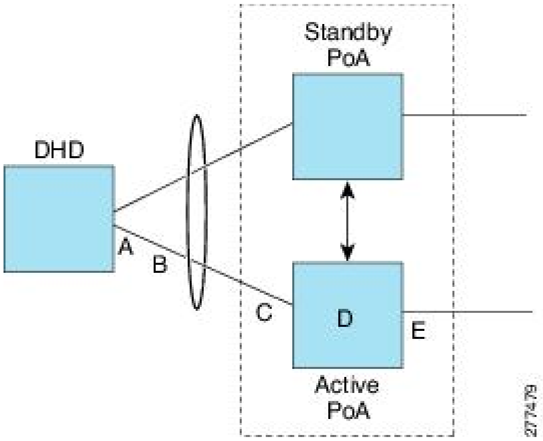
Port Key

The port key indicates which links can form a LAG on a given system. The key is locally significant to an LACP system and need not match the key on an LACP peer. Two links are candidates to join the same LAG if they have the same key on the DHD and the same key on the PoAs; however, the key on the DHD is not required to be the same as the key on the PoAs. Given that the key is configured according to the need to aggregate ports, there are no special considerations for this parameter for multichassis operation.

Failure Protection Scenarios

The mLACP feature provides network resiliency by protecting against port, link, and node failures. These failures can be categorized into five types. The figure below shows the failure points in a network, denoted by the letters A through E.

- A--Failure of the uplink port on the DHD
- B--Failure of the Ethernet link
- C--Failure of the downlink port on the active PoA
- D--Failure of the active PoA node
- E--Failure of the active PoA uplinks



When any of these faults occur, the system reacts by triggering a switchover from the active PoA to the standby PoA. The switchover involves failing over the PoA’s uplinks and downlinks simultaneously.

Failure points A and C are port failures. Failure point B is an Ethernet link failure and failure point D is a node failure. Failure point E can represent one of four different types of uplink failures when the PoAs connect to an MPLS network:

- Pseudowire failure--Monitoring individual pseudowires (for example, using VCCV-BFD) and, upon a pseudowire failure, declare uplink failure for the associated service instances.
- Remote PE IP path failure--Monitoring the IP reachability to the remote PE (for example, using IP Route-Watch) and, upon route failure, declare uplink failure for all associated service instances.
- LSP failure--Monitoring the LSP to a given remote PE (for example, using automated LSP-Ping) and, upon LSP failure, declare uplink failure for all associated service instances.
- PE isolation--Monitoring the physical core-facing interfaces of the PE. When all of these interfaces go down, the PE effectively becomes isolated from the core network, and the uplink failure is declared for all affected service instances.

As long as the IP/MPLS network employs native redundancy and resiliency mechanisms such as MPLS fast reroute (FRR), the mLACP solution is sufficient for providing protection against PE isolation. Pseudowire, LSP, and IP path failures are managed by the native IP/MPLS protection procedures. That is, interchassis failover via mLACP is triggered only when a PE is completely isolated from the core network, because native

IP/MPLS protection mechanisms are rendered useless. Therefore, failure point E is used to denote PE isolation from the core network.

Note The set of core-facing interfaces that should be monitored are identified by explicit configuration. The set of core-facing interfaces must be defined independently per redundancy group. Failure point E (unlike failure point A, B, or C) affects and triggers failover for all the multichassis LAGs configured on a given PoA.

Operational Variants

LACP provides a mechanism by which a set of one or more links within a LAG are placed in standby mode to provide link redundancy between the devices. This redundancy is normally achieved by configuring more ports with the same key than the number of links a device can aggregate in a given LAG (due to hardware or software restrictions, or due to configuration). For example, for active/standby redundancy, two ports are configured with the same port key, and the maximum number of allowed links in a LAG is configured to be 1. If the DHD and PoAs are all capable of restricting the number of links per LAG by configuration, three operational variants are possible.

DHD-based Control

The value of PoAs must be greater than the value of DHD. In DHD-based control, maximum number of links per bundle should be one. The PoAs must be configured to limit the maximum number of links per bundle to be greater than one. Thus, the selection of the active/standby link is the responsibility of the DHD. Which link is designated active and which is marked standby depends on the relative port priority, as configured on the system with the higher system priority. A PoA configured with a higher system priority can still determine the selection outcome. The DHD makes the selection and places the link with lower port priority in standby mode.

To accommodate DHD-controlled failover, the DHD must be configured with the max-bundle value equal to a number of links (L), where L is the fewest number of links connecting the DHD to a PoA. The max-bundle value restricts the DHD from bundling links to both PoAs at the same time (active/active). Although the DHD controls the selection of active/standby links, the PoA can still dictate the individual member link priorities by configuring the PoA’s virtual LACP instance with a lower system priority value than the DHD’s system priority.

The DHD control variant must be used with a PoA minimum link threshold failure policy where the threshold is set to L (same value for L as described above). A minimum link threshold must be configured on each of the PoAs because an A, B, or C link failure that does not trigger a failover (minimum link threshold is still satisfied) causes the DHD to add one of the standby links going to the standby PoA to the bundle. This added link results in the unsupported active/active scenario.

Note DHD control does not use the mLACP hot-standby state on the standby PoA, which results in higher failover times than the other variants.

DHD control eliminates the split brain problem on the attachment circuit side by limiting the DHD’s attempts to bundle all the links.

PoA Control

In PoA control, the PoA is configured to limit the maximum number of links per bundle to be equal to the number of links (L) going to the PoA. The DHD is configured with that parameter set to some value greater than L. Thus, the selection of the active/standby links becomes the responsibility of the PoA.

Shared Control (PoA and DHD)

In shared control, both the DHD and the PoA are configured to limit the maximum number of links per bundle to L--the number of links going to the PoA. In this configuration, each device independently selects the active/standby link. Shared control is advantageous in that it limits the split-brain problem in the same manner as DHD control, and shared control is not susceptible to the active/active tendencies that are prevalent in DHD control. A disadvantage of shared control is that the failover time is determined by both the DHD and the PoA, each changing the standby links to SELECTED and waiting for each of the WAIT_WHILE_TIMERs to expire before moving the links to IN_SYNC. The independent determination of failover time and change of link states means that both the DHD and PoAs need to support the LACP fast-switchover feature in order to provide a failover time of less than one second.

mLACP Failover

The mLACP forces a PoA failover to the standby PoA when one of the following failures occurs:

- Failure of the DHD uplink port, Ethernet link, or downlink port on the active PoA—A policy failover is triggered via a configured failover policy and is considered a forced failover. When the number of active and SELECTED links to the active PoA goes below the configured minimum threshold, mLACP forces a failover to the standby PoA's member links. This minimum threshold is configured using the **lacp min-links** command in interface configuration mode. The PoAs determine the failover independent of the operational control variant in use.

- Failure of the active PoA—This failure is detected by the standby PoA. mLACP automatically fails over to standby because mLACP on the standby PoA is notified of failure via ICRM and brings up its local member links. In the DHD-controlled variant, this failure looks the same as a total member link failure, and the DHD activates the standby links.

- Failure of the active PoA uplinks—mLACP is notified by ICRM of PE isolation and relinquishes its active member links. This failure is a "forced failover" and is determined by the PoAs independent of the operational control variant in use.

Dynamic Port Priority

The default failover mechanism uses dynamic port priority changes on the local member links to force the LACP selection logic to move the required standby link(s) to the SELECTED and Collecting_Distributing state. This state change occurs when the LACP actor port priority values for all affected member links on the currently active PoA are changed to a higher numeric value than the standby PoA's port priority (which gives the standby PoA ports a higher claim to bundle links). Changing the actor port priority triggers the transmission of an mLACP Port Config Type-Length-Value (TLV) message to all peers in the redundancy group. These messages also serve as notification to the standby PoA(s) that the currently active PoA is attempting to relinquish its role. The LACP then transitions the standby link(s) to the SELECTED state and moves all the currently active links to STANDBY.

Dynamic port priority changes are not automatically written back to the running configuration or to the NVRAM configuration. If you want the current priorities to be used when the system reloads, the **mlacp lag-priority** command must be used and the configuration must be saved.

Revertive and Nonrevertive Modes

Dynamic port priority functionality is used by the mLACP feature to provide both revertive mode and nonrevertive mode. The default operation is revertive, which is the default behavior in single chassis LACP. Nonrevertive mode can be enabled on a per port-channel basis by using the **lacp failover non-revertive**command in interface configuration mode. In Cisco IOS Release 12.2(33)SRE this command is supported only for mLACP.

Nonrevertive mode is used to limit failover and, therefore, possible traffic loss. Dynamic port priority changes are utilized to ensure that the newly activated PoA remains active after the failed PoA recovers.

Revertive mode operation forces the configured primary PoA to return to active state after it recovers from a failure. Dynamic port priority changes are utilized when necessary to allow the recovering PoA to resume its active role.

Brute Force Shutdown

A brute-force shutdown is a forced failover mechanism to bring down the active physical member link interface(s) for the given LAG on the PoA that is surrendering its active status. This mechanism does not depend on the DHD's ability to manage dynamic port priority changes and compensates for deficiencies in the DHD's LACP implementation.

The brute-force shutdown changes the status of each member link to ADMIN_DOWN to force the transition of the standby links to the active state. Note that this process eliminates the ability of the local LACP implementation to monitor the link state.

The brute-force shutdown operates in revertive mode, so dynamic port priorities cannot be used to control active selection. The brute-force approach is configured by the **lacp failover brute-force** command in interface configuration mode. This command is not allowed in conjunction with a nonrevertive configuration.

Peer Monitoring with Interchassis Redundancy Manager

There are two ways in which a peer can be monitored with Interchassis Redundancy Manager (ICRM):

- Routewatch (RW)--This method is the default.
- Bidirectional Forwarding Detection (BFD)--You must configure the redundancy group with the **monitor peer bfd** command.

Note For stateful switchover (SSO) deployments (with redundant support in the chassis), BFD monitoring and a static route for the ICCP connection are required to prevent "split brain" after an SSO failover.

For each redundancy group, for each peer (member IP), a monitoring adjacency is created. If there are two peers with the same IP address, the adjacency is shared regardless of the monitoring mode. For example, if redundancy groups 1 and 2 are peered with member IP 10.10.10.10, there is only one adjacency to 10.10.10.10, which is shared in both redundancy groups. Furthermore, redundancy group 1 can use BFD monitoring while redundancy group 2 is using RW.

Note BFD is completely dependent on RW--there must be a route to the peer for ICRM to initiate BFD monitoring. BFD implies RW and sometimes the status of the adjacency may seem misleading but is accurately representing the state. Also, if the route to the peer PoA is not through the directly connected (back-to-back) link between the systems, BFD can give misleading results.

An example of output from the **show redundancy interchassis** command follows:

```
Device# show redundancy interchassis
Redundancy Group 1 (0x1)
  Applications connected: mLACP
  Monitor mode: Route-watch
  member ip: 201.0.0.1 ‘mlacp-201’, CONNECTED
    Route-watch for 201.0.0.1 is UP
    mLACP state: CONNECTED
ICRM fast-failure detection neighbor table
  IP Address       Status Type Next-hop IP      Interface
  ==========       ====== ==== ===========      =========
  201.0.0.1        UP     RW
```

To interpret the adjacency status displayed by the **show redundancy interchassis** command, refer to the table below.

Table 26: Status Information from the show redundancy interchassis command

Adjacency Type	Adjacency Status	Meaning
RW	DOWN	RW or BFD is configured, but there is no route for the given IP address.
RW	UP	RW or BFD is configured. RW is up, meaning there is a valid route to the peer. If BFD is configured and the adjacency status is UP, BFD is probably not configured on the interface of the route’s adjacency.
BFD	DOWN	BFD is configured. A route exists and the route’s adjacency is to an interface that has BFD enabled. BFD is started but the peer is down. The DOWN status can be because the peer is not present or BFD is not configured on the peer’s interface.
BFD	UP	BFD is configured and operational.

Note If the adjacency type is “BFD,” RW is UP regardless of the BFD status.

MAC Flushing Mechanisms

When mLACP is used to provide multichassis redundancy in multipoint bridged services (for example, VPLS), there must be a MAC flushing notification mechanism in order to prevent potential traffic failure.

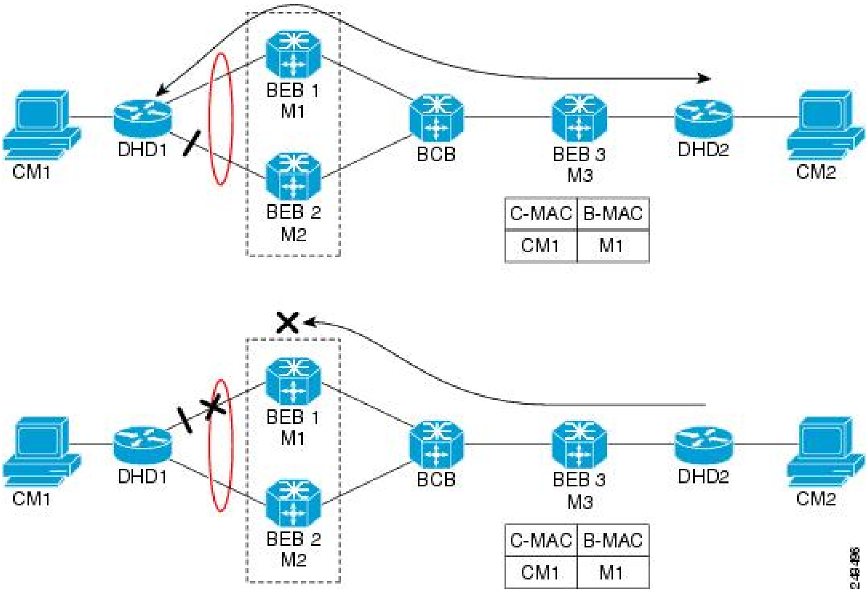
At the failover from a primary PoA to a secondary PoA, a service experiences traffic failure when the DHD in question remains inactive and while other remote devices in the network are attempting to send traffic to that DHD. Remote bridges in the network have stale MAC entries pointing to the failed PoA and direct traffic destined to the DHD to the failed PoA, where the traffic is dropped. This failure continues until the remote devices age out their stale MAC address table entries (which typically takes five minutes). To prevent this anomaly, the newly active PoA, which has taken control of the service, transmits a MAC flush notification message to the remote devices in the network to flush their stale MAC address entries for the service in question.

The exact format of the MAC flushing message depends on the nature of the network transport: native 802.1Q/802.1ad Ethernet, native 802.1ah Ethernet, VPLS, or provider backbone bridge (PBB) over VPLS. Furthermore, in the context of 802.1ah, it is important to recognize the difference between mechanisms used for customer-MAC (C-MAC) address flushing versus bridge-MAC (B-MAC) address flushing.

The details of the various mechanisms are discussed in the following sections.

Multiple I-SID Registration Protocol

Multiple I-SID Registration Protocol (MIRP) is enabled by default on 802.1ah service instances. The use of MIRP in 802.1ah networks is shown in the figure below.



Device DHD1 is dual-homed to two 802.1ah backbone edge bridges (BEB1 and BEB2). Assume that initially the primary path is through BEB1. In this configuration BEB3 learns that the host behind DHD1 (with MAC address CM1) is reachable via the destination B-MAC M1. If the link between DHD1 and BEB1 fails and the host behind DHD1 remains inactive, the MAC cache tables on BEB3 still refer to the BEB1 MAC address even though the new path is now via BEB2 with B-MAC address M2. Any bridged traffic destined from the host behind DHD2 to the host behind DHD1 is wrongfully encapsulated with B-MAC M1 and sent over the MAC tunnel to BEB1, where the traffic fails.

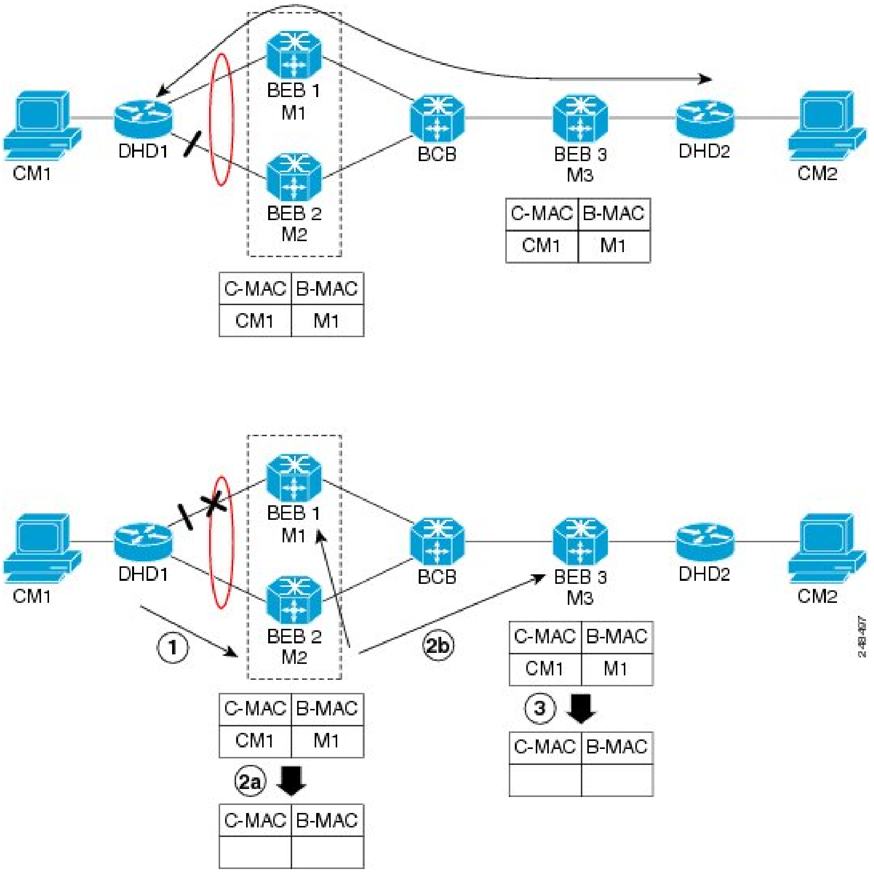
To circumvent the traffic failure problem when the link between DHD1 and BEB1 fails, BEB2 performs two tasks:

- Flushes its own MAC address table for the service or services in question.
- Transmits an MIRP message on its uplink to signal the far end BEB (BEB3) to flush its MAC address table. Note that the MIRP message is transparent to the backbone core bridges (BCBs). The MIRP message is processed on a BEB because only BCBs learn and forward based on B-MAC addresses and they are transparent to C-MAC addresses.

Note

MIRP triggers C-MAC address flushing for both native 802.1ah and PBB over VPLS.

The figure below shows the operation of the MIRP.



The MIRP has not been defined in IEEE but is expected to be based on the IEEE 802.1ak Multiple Registration Protocol (MRP). MRP maintains a complex finite state machine (FSM) for generic attribute registration. In the case of MIRP, the attribute is an I-SID. As such, MIRP provides a mechanism for BEBs to build and prune a per I-SID multicast tree. The C-MAC flushing notification capability of MIRP is a special case of attribute registration in which the device indicates that an MIRP declaration is “new,” meaning that this notification is the first time a BEB is declaring interest in a particular I-SID.

LDP MAC Address Withdraw

When the mLACP feature is used for PE redundancy in traditional VPLS (that is, not PBB over VPLS), the MAC flushing mechanism is based on the LDP MAC Address Withdraw message as defined in RFC 4762.

The required functional behavior is as follows: Upon a failover from the primary PoA to the standby PoA, the standby PoA flushes its local MAC address table for the affected services and generates the LDP MAC Address Withdraw messages to notify the remote PEs to flush their own MAC address tables. One message is generated for each pseudowire in the affected virtual forwarding instances (VFIs).

How to Configure mLACP

Configuring Interchassis Group and Basic mLACP Commands (Global Redundancy Group Configuration)

Perform this task to set up the communication between multiple PoAs and to configure them in the same group.

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 **redundancy**

Example:

```
Router(config)# redundancy
```

Enters redundancy configuration mode.

Step 4 **interchassis group** *group-id*

Example:

```
Router(config-red)# interchassis group 50
```

Configures an interchassis group within the redundancy configuration mode and enters interchassis redundancy mode.

Step 5 **monitor peer bfd**

Example:

```
Router(config-r-ic)# monitor peer bfd
```

Configures the BFD option to monitor the state of the peer. The default option is route-watch.

Step 6 **member ip** *ip-address*

Example:

```
Router(config-r-ic)# member ip 172.3.3.3
```

Configures the IP address of the mLACP peer member group.

Step 7 **mlacp node-id** *node-id*

Example:

```
Router(config-r-ic)# mlacp node-id 5
```

Defines the node ID used in the LACP Port ID field by this member of the mLACP redundancy group.

- The valid range is 0 to 7, and the value should be different from the peer values.

Step 8 **mlacp system-mac** *mac-address*

Example:

```
Router(config-r-ic)# mlacp system-mac aa12.be45.d799
```

Defines and advertises the system MAC address value to the mLACP members of the redundancy group for arbitration.

- The format of the *mac-address* argument must be in standard MAC address format: aabb.ccdd.eeff.

Step 9 **mlacp system-priority** *priority-value*

Example:

```
Router(config-r-ic)# mlacp system-priority 100
```

Defines the system priority advertised to the other mLACP members of the redundancy group.

- System priority values are 1 to 65535. Default value is 32768.
- The assigned values should be lower than the DHD.

Step 10 **backbone interface** *type number*

Example:

```
Router(config-r-ic)#
backbone interface GigabitEthernet2/3
```

Defines the backbone interface for the mLACP configuration.

Step 11 **end**

Example:

```
Router(config-r-ic)# end
```

Returns the CLI to privileged EXEC mode.

Configuring the mLACP Interchassis Group and Other Port-Channel Commands

Perform this task to set up mLACP attributes specific to a port channel. The **mlacp interchassis group** command links the port-channel interface to the interchassis group that was created in the previous Configuring Interchassis Group and Basic mLACP Commands (Global Redundancy Group Configuration) , on page 327.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel- number*
4. **lacp max-bundle** *max-bundles*
5. **lacp failover** {**brute-force**| **non-revertive**}
6. **exit**
7. **redundancy**
8. **interchassis group** *group-id*
9. **exit**
10. **exit**
11. **errdisable recovery cause mlacp-minlink**
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 4	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 4`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles* argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.

	Command or Action	Purpose

Step 5	**lacp failover** {**brute-force**| **non-revertive**}
	Example:
	`Router(config-if)# lacp failover brute-force`
		Sets the mLACP switchover to nonrevertive or brute force. This command is optional.
		• Default value is revertive (with 180-second delay).
		• If you configure brute force, a minimum link failure for every mLACP failure occurs or the dynamic lag priority value is modified.

Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 7	**redundancy** **Example:** `Router(config)# redundancy`	Enters redundancy configuration mode.
Step 8	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port channel. The *group-id* should match the configured redundancy group.
Step 9	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.
Step 10	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 11	**errdisable recovery cause mlacp-minlink** **Example:** `Router(config)# errdisable recovery cause mlacp-minlink`	Enables automatic recovery from a failover state of the port channel.

Step 12	**end**
	Example:
	`Router(config)# end`
		Returns the CLI to privileged EXEC mode.

Configuring Redundancy for VPWS

Perform this task to provide Layer 2 VPN service redundancy for VPWS.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. **encapsulation mpls**
5. **status peer topology dual-homed**
6. **exit**
7. **interface port-channel** *port-channel-number*
8. **no ip address**
9. **lacp fast-switchover**
10. **lacp max-bundle** *max-bundles*
11. **exit**
12. **redundancy**
13. **interchassis group** *group-id*
14. **exit**
15. **exit**
16. **interface port-channel** *port-channel-number*
17. **service instance** *id* **ethernet** [*evc-name*]
18. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**]
19. **exit**
20. **xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}]
21. **backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*]
22. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *pw-class-name* **Example:** `Router(config)# pseudowire-class ether-pw`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.

	Command or Action	Purpose
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	**status peer topology dual-homed** **Example:** `Router(config-pw-class)# status peer topology dual-homed`	Enables the reflection of the attachment circuit status onto both the primary and secondary pseudowires. This condition is necessary if the peer PEs are connected to a dual-homed device.
Step 6	**exit** **Example:** `Router(config-pw-class)# exit`	Exits pseudowire class configuration mode.
Step 7	**interface port-channel** *port-channel-number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 8	**no ip address** **Example:** `Router(config-if)# no ip address`	Specifies that the VLAN interface does not have an IP address assigned to it.
Step 9	**lacp fast-switchover** **Example:** `Router(config-if)# lacp fast-switchover`	Enables LACP 1-to-1 link redundancy.
Step 10	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 4`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles*argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.
Step 11	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 12	**redundancy** **Example:**	Enters redundancy configuration mode.

	Command or Action	Purpose
	`Router(config)# redundancy`	
Step 13	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port channel. - The *group-id* should match the configured redundancy group.
Step 14	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.
Step 15	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 16	**interface port-channel** *port-channel-number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 17	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance.
Step 18	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 19	**exit** **Example:** `Router(config-if-srv)# exit`	Exits service instance configuration mode.
Step 20	**xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}] **Example:** `Router(config-if)# xconnect 10.0.3.201 123 pw-class ether-pw`	Binds an attachment circuit to a pseudowire.
Step 21	**backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*]	Specifies a redundant peer for a pseudowire virtual circuit.

	Command or Action	Purpose
	Example: `Router(config-if)# backup peer 10.1.1.1 123 pw-class ether-pw`	

Step 22	**end**
	Example:
	`Router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Configuring Redundancy for VPWS on ME3600 Series Switches

Perform this task to provide Layer 2 VPN service redundancy for VPWS on Cisco ME3600, ME3600X 24CX, ME3800 series switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. **encapsulation mpls**
5. **status peer topology dual-homed**
6. **exit**
7. **interface port-channel** *port-channel-number*
8. **switchport mode trunk**
9. **switchport trunk allowed vlan none**
10. **lacp fast-switchover**
11. **lacp max-bundle** *max-bundles*
12. **exit**
13. **redundancy**
14. **interchassis group** *group-id*
15. **exit**
16. **exit**
17. **interface port-channel** *port-channel-number*
18. **service instance** *id* **ethernet** [*evc-name*]
19. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**]
20. **exit**
21. **xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}]
22. **backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*]
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *pw-class-name* **Example:** `Router(config)# pseudowire-class ether-pw`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	**status peer topology dual-homed** **Example:** `Router(config-pw-class)# status peer topology dual-homed`	Enables the reflection of the attachment circuit status onto both the primary and secondary pseudowires. This condition is necessary if the peer PEs are connected to a dual-homed device.
Step 6	**exit** **Example:** `Router(config-pw-class)# exit`	Exits pseudowire class configuration mode.
Step 7	**interface port-channel** *port-channel-number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 8	**switchport mode trunk** **Example:** `Router(config-if)# switchport mode trunk`	Specifies the port channel as trunking VLAN Layer 2 interface.
Step 9	**switchport trunk allowed vlan none** **Example:** `Router(config-if)# switchport trunk allowed vlan none`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.

	Command or Action	Purpose
Step 10	**lacp fast-switchover** **Example:** `Router(config-if)# lacp fast-switchover`	Enables LACP 1-to-1 link redundancy.
Step 11	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 4`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles*argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.
Step 12	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 13	**redundancy** **Example:** `Router(config)# redundancy`	Enters redundancy configuration mode.
Step 14	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port channel. • The *group-id* should match the configured redundancy group.
Step 15	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.
Step 16	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 17	**interface port-channel** *port-channel-number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 18	**service instance** *id* **ethernet** [*evc-name*] **Example:**	Configures an Ethernet service instance.

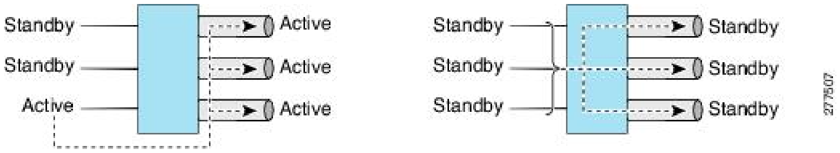
	Command or Action	Purpose
	`Router(config-if)# service instance 1 ethernet`	
Step 19	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 20	**exit** **Example:** `Router(config-if-srv)# exit`	Exits service instance configuration mode.
Step 21	**xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}] **Example:** `Router(config-if)# xconnect 10.0.3.201 123 pw-class ether-pw`	Binds an attachment circuit to a pseudowire.
Step 22	**backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*] **Example:** `Router(config-if)# backup peer 10.1.1.1 123 pw-class ether-pw`	Specifies a redundant peer for a pseudowire virtual circuit.

Step 23	**end**
	Example:
	`Router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

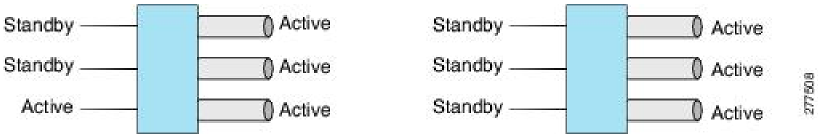
Configuring Redundancy for VPLS

Coupled and Decoupled Modes for VPLS

VPLS can be configured in either coupled mode or decoupled mode. Coupled mode is when at least one attachment circuit in VFI changes state to active, all pseudowires in VFI advertise active. When all attachment circuits in VFI change state to standby, all pseudowires in VFI advertise standby mode. See the figure below.



VPLS decoupled mode is when all pseudowires in the VFI are always active and the attachment circuit state is independent of the pseudowire state. This mode provides faster switchover time when a platform does not support pseudowire status functionality, but extra flooding and multicast traffic will be dropped on the PE with standby attachment circuits. However, if the attachment circuit is down, all pseudowires also go down. See the figure below.



Steps for Configuring Redundancy for VPLS

Perform the following task to configure redundancy for VPLS.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **manual**
4. **vpn id** *vpn-id*
5. **status decoupled**
6. **neighbor** *neighbor ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
7. **exit**
8. **interface port-channel** *port-channel- number*
9. **no ip address**
10. **lacp fast-switchover**
11. **lacp max-bundle** *max-bundles*
12. **exit**
13. **redundancy**
14. **interchassis group** *group-id*
15. **exit**
16. **exit**
17. **interface port-channel** *port-channel- number*
18. **service instance** *id* **ethernet** [*evc-name*]
19. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**]
20. **bridge-domain** *bridge-id* [**split-horizon** [**group** *group-id*]]
21. **exit**
22. **interface vlan** *vlanid*
23. **no ip address**
24. **xconnect vfi** *vfi-name*
25. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *name* **manual** **Example:** `Router(config)# l2 vfi vfi1 manual`	Establishes a Layer 2 VFI between two separate networks and enters VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Router(config-vfi)# vpn id 100`	Sets or updates a Virtual Private Network (VPN) ID on a VPN routing and forwarding (VRF) instance.
Step 5	**status decoupled** **Example:** `Router(config-vfi)# status decoupled`	(Optional) Enables decoupled mode. The state of the attachment circuits on the user-facing Provider Edge (uPE) is decoupled from the state of the pseudowires. The mLACP controls the state of the attachment circuits.

Step 6	**neighbor** *neighbor ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
	Example:
	`Router(config-vfi)# neighbor 10.1.1.1 50 encapsulation mpls`
		Specifies the routers that should form a VFI connection.
		• Repeat this command for each neighbor.

Step 7	**exit** **Example:** `Router(config-vfi)# exit`	Exits VFI configuration mode and returns to global configuration mode.
Step 8	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 9	**no ip address** **Example:** `Router(config-if)# no ip address`	Specifies that the VLAN interface does not have an IP address assigned to it.
Step 10	**lacp fast-switchover** **Example:**	Enables LACP 1-to-1 link redundancy.

	Command or Action	Purpose
	`Router(config-if)# lacp fast-switchover`	
Step 11	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 2`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles*argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.
Step 12	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 13	**redundancy** **Example:** `Router(config)# redundancy`	• Enters redundancy configuration mode.
Step 14	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port-channel. • The *group-id* should match the configured redundancy group.
Step 15	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.
Step 16	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 17	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 18	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance and enters Ethernet service configuration mode.

	Command or Action	Purpose
Step 19	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 20	**bridge-domain** *bridge-id* [**split-horizon** [**group** *group-id*]] **Example:** `Router(config-if-srv)# bridge-domain 200`	Configures the bridge domain. Binds the service instance to a bridge domain instance where *domain-number* is the identifier for the bridge domain instance.
Step 21	**exit** **Example:** `Router(config-if-srv)# exit`	Exits service instance configuration mode.
Step 22	**interface vlan** *vlanid* **Example:** `Router(config-if)# interface vlan 200`	Creates a dynamic switch virtual interface (SVI).
Step 23	**no ip address** **Example:** `Router(config-if)# no ip address`	Specifies that the VLAN interface does not have an IP address assigned to it.
Step 24	**xconnect vfi** *vfi-name* **Example:** `Router(config-if)# xconnect vfi vfi-16`	Specifies the Layer 2 VFI that you are binding to the VLAN port.

Step 25	**end**
	Example:
	`Router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Steps for Configuring Redundancy for VPLS on ME3600 Series Switches

Perform the following task to configure redundancy for VPLS on Cisco ME3600, ME3600X 24CX, ME3800 series switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **manual**

4. **vpn id** *vpn-id*
5. **status decoupled**
6. **neighbor** *neighbor ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
7. **exit**
8. **interface port-channel** *port-channel- number*
9. **switchport mode trunk**
10. **switchport trunk allowed vlan none**
11. **lacp fast-switchover**
12. **lacp max-bundle** *max-bundles*
13. **exit**
14. **redundancy**
15. **interchassis group** *group-id*
16. **exit**
17. **exit**
18. **interface port-channel** *port-channel- number*
19. **service instance** *id* **ethernet** [*evc-name*]
20. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**]
21. **bridge-domain** *bridge-id* [**split-horizon** [**group** *group-id*]]
22. **exit**
23. **interface vlan** *vlanid*
24. **no ip address**
25. **xconnect vfi** *vfi-name*
26. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *name* **manual** **Example:** `Router(config)# l2 vfi vfi1 manual`	Establishes a Layer 2 VFI between two separate networks and enters VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Router(config-vfi)# vpn id 100`	Sets or updates a Virtual Private Network (VPN) ID on a VPN routing and forwarding (VRF) instance.

	Command or Action	Purpose
Step 5	**status decoupled** **Example:** `Router(config-vfi)# status decoupled`	(Optional) Enables decoupled mode. The state of the attachment circuits on the user-facing Provider Edge (uPE) is decoupled from the state of the pseudowires. The mLACP controls the state of the attachment circuits.

Step 6	**neighbor** *neighbor ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*}
	Example:
	`Router(config-vfi)# neighbor 10.1.1.1 50 encapsulation mpls`
		Specifies the routers that should form a VFI connection.
		• Repeat this command for each neighbor.

Step 7	**exit** **Example:** `Router(config-vfi)# exit`	Exits VFI configuration mode and returns to global configuration mode.
Step 8	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 9	**switchport mode trunk** **Example:** `Router(config-if)# switchport mode trunk`	Specifies the port channel as trunking VLAN Layer 2 interface.
Step 10	**switchport trunk allowed vlan none** **Example:** `Router(config-if)# switchport trunk allowed vlan none`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.
Step 11	**lacp fast-switchover** **Example:** `Router(config-if)# lacp fast-switchover`	Enables LACP 1-to-1 link redundancy.
Step 12	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 2`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles*argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.

	Command or Action	Purpose
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 14	**redundancy** **Example:** `Router(config)# redundancy`	• Enters redundancy configuration mode.
Step 15	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port-channel. • The *group-id* should match the configured redundancy group.
Step 16	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.
Step 17	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 18	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 19	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance and enters Ethernet service configuration mode.
Step 20	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 21	**bridge-domain** *bridge-id* [**split-horizon** [**group** *group-id*]] **Example:** `Router(config-if-srv)# bridge-domain 200`	Configures the bridge domain. Binds the service instance to a bridge domain instance where *domain-number* is the identifier for the bridge domain instance.

	Command or Action	Purpose
Step 22	**exit** **Example:** `Router(config-if-srv)# exit`	Exits service instance configuration mode.
Step 23	**interface vlan** *vlanid* **Example:** `Router(config-if)# interface vlan 200`	Creates a dynamic switch virtual interface (SVI).
Step 24	**no ip address** **Example:** `Router(config-if)# no ip address`	Specifies that the VLAN interface does not have an IP address assigned to it.
Step 25	**xconnect vfi** *vfi-name* **Example:** `Router(config-if)# xconnect vfi vfi-16`	Specifies the Layer 2 VFI that you are binding to the VLAN port.

Step 26	**end**
	Example:
	`Router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Configuring Hierarchical VPLS

Perform this task to configure Hierarchical VPLS (H-VPLS).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. **encapsulation mpls**
5. **status peer topology dual-homed**
6. **status decoupled**
7. **exit**
8. **interface port-channel** *port-channel- number*
9. **no ip address**
10. **lacp fast-switchover**
11. **lacp max-bundle** *max-bundles*
12. **exit**
13. **redundancy**
14. **interchassis group** *group-id*

15. **exit**
16. **exit**
17. **interface port-channel** *port-channel- number*
18. **service instance** *id* **ethernet** [*evc-name*]
19. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**]
20. **exit**
21. **xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}]
22. **backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*]
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *pw-class-name* **Example:** `Router(config)# pseudowire-class ether-pw`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	**status peer topology dual-homed** **Example:** `Router(config-pw-class)# status peer topology dual-homed`	Enables the reflection of the attachment circuit status onto both the primary and secondary pseudowires. This configuration is necessary if the peer PEs are connected to a dual-homed device.
Step 6	**status decoupled** **Example:** `Router(config-pw-class)# status decoupled`	(Optional) Enables decoupled mode. The state of the attachment circuits on the uPE is decoupled from the state of the pseudowires. The mLACP controls the state of the attachment circuits.
Step 7	**exit** **Example:**	Exits pseudowire class configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Router(config-pw-class)# exit`	
Step 8	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 9	**no ip address** **Example:** `Router(config-if)# no ip address`	Specifies that the VLAN interface does not have an IP address assigned to it.
Step 10	**lacp fast-switchover** **Example:** `Router(config-if)# lacp fast-switchover`	Enables LACP 1-to-1 link redundancy.
Step 11	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 4`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles*argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.
Step 12	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 13	**redundancy** **Example:** `Router(config)# redundancy`	Enters redundancy configuration mode.
Step 14	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port channel. • The *group-id* should match the configured redundancy group.
Step 15	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.

	Command or Action	Purpose
Step 16	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 17	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 18	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance and enters Ethernet service configuration mode.
Step 19	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 20	**exit** **Example:** `Router(config-if-srv)# exit`	Exits service instance configuration mode.

Step 21	**xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}]
	Example:
	`Router(config-if)# xconnect 10.0.3.201 123`
	`pw-class vlan-xconnect`
		Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.

Step 22	**backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*]
	Example:
	`Router(config-if)# backup peer 10.1.1.1 123`
	`pw-class ether-pw`
		Specifies a redundant peer for a pseudowire virtual circuit.

Step 23	**end**
	Example:
	`Router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Configuring Hierarchical VPLS on ME3600 Series Switches

Perform this task to configure Hierarchical VPLS (H-VPLS) on Cisco ME3600, ME3600X 24CX, ME3800 series switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. **encapsulation mpls**
5. **status peer topology dual-homed**
6. **status decoupled**
7. **exit**
8. **interface port-channel** *port-channel- number*
9. **switchport mode trunk**
10. **switchport trunk allowed vlan none**
11. **lacp fast-switchover**
12. **lacp max-bundle** *max-bundles*
13. **exit**
14. **redundancy**
15. **interchassis group** *group-id*
16. **exit**
17. **exit**
18. **interface port-channel** *port-channel- number*
19. **service instance** *id* **ethernet** [*evc-name*]
20. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**]
21. **exit**
22. **xconnect** *peer-ip-address vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}]
23. **backup peer** *peer-router-ip-addr vcid* [**pw-class** *pw-class-name*] [**priority** *value*]
24. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**pseudowire-class** *pw-class-name* **Example:** `Router(config)# pseudowire-class ether-pw`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	**status peer topology dual-homed** **Example:** `Router(config-pw-class)# status peer topology dual-homed`	Enables the reflection of the attachment circuit status onto both the primary and secondary pseudowires. This configuration is necessary if the peer PEs are connected to a dual-homed device.
Step 6	**status decoupled** **Example:** `Router(config-pw-class)# status decoupled`	(Optional) Enables decoupled mode. The state of the attachment circuits on the uPE is decoupled from the state of the pseudowires. The mLACP controls the state of the attachment circuits.
Step 7	**exit** **Example:** `Router(config-pw-class)# exit`	Exits pseudowire class configuration mode and returns to global configuration mode.
Step 8	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 9	**switchport mode trunk** **Example:** `Router(config-if)# switchport mode trunk`	Specifies the port channel as trunking VLAN Layer 2 interface.
Step 10	**switchport trunk allowed vlan none** **Example:** `Router(config-if)# switchport trunk allowed vlan none`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.
Step 11	**lacp fast-switchover** **Example:** `Router(config-if)# lacp fast-switchover`	Enables LACP 1-to-1 link redundancy.

	Command or Action	Purpose
Step 12	**lacp max-bundle** *max-bundles* **Example:** `Router(config-if)# lacp max-bundle 4`	Configures the max-bundle links that are connected to the PoA. The value of the *max-bundles*argument should not be less than the total number of links in the LAG that are connected to the PoA. • Determines whether the redundancy group is under DHD control, PoA control, or both. • Range is 1 to 8. Default value is 8.
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 14	**redundancy** **Example:** `Router(config)# redundancy`	Enters redundancy configuration mode.
Step 15	**interchassis group** *group-id* **Example:** `Router(config-red)# interchassis group 230`	Specifies that the port channel is an mLACP port channel. • The *group-id* should match the configured redundancy group.
Step 16	**exit** **Example:** `Router(config-r-ic)# exit`	Exits interchassis redundancy mode.
Step 17	**exit** **Example:** `Router(config-red)# exit`	Exits redundancy configuration mode.
Step 18	**interface port-channel** *port-channel- number* **Example:** `Router(config)# interface port-channel1`	Configures the port channel and enters interface configuration mode.
Step 19	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance and enters Ethernet service configuration mode.
Step 20	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:**	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.

	Command or Action	Purpose
	`Router(config-if-srv)# encapsulation dot1q 100`	
Step 21	**exit** **Example:** `Router(config-if-srv)# exit`	Exits service instance configuration mode.
Step 22	**xconnect** *peer-ip-address* *vc-id* {**encapsulation mpls** | **pw-class** *pw-class-name*} [**pw-class** *pw-class-name*] [**sequencing** {**transmit** | **receive** | **both**}] **Example:** `Router(config-if)# xconnect 10.0.3.201 123 pw-class vlan-xconnect`	Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.
Step 23	**backup peer** *peer-router-ip-addr* *vcid* [**pw-class** *pw-class-name*] [**priority** *value*] **Example:** `Router(config-if)# backup peer 10.1.1.1 123 pw-class ether-pw`	Specifies a redundant peer for a pseudowire virtual circuit.

Step 24	**end**
	Example:
	`Router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Troubleshooting mLACP

Debugging mLACP

Use these **debug** commands for general mLACP troubleshooting.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug redundancy interchassis** {**all** | **application** | **error** | **event** | **monitor**}

Example:

```
Router# debug redundancy interchassis all
```

- Enables debugging of the interchassis redundancy manager.

Step 3 **debug mpls ldp iccp**

Example:

```
Router# debug mpls ldp iccp
```

- Enables debugging of the InterChassis Control Protocol (ICCP).

Step 4 **debug lacp** [**all** | **event**| **fsm**| **misc**| **multi-chassis** [**all** | **database** | **lacp-mgr** | **redundancy-group** | **user-interface**] | **packet**]

Example:

```
Router# debug lacp multi-chassis all
```

Enables debugging of LACP activity.

- This command is run on the switch processor.

Step 5 **debug lacp etherchannel**

Example:

```
Router# debug lacp etherchannel
```

Enables debugging for etherchannel component.

Debugging mLACP on an Attachment Circuit or EVC

Use these **debug** commands for troubleshooting mLACP on an attachment circuit or on an EVC.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug acircuit** {**checkpoint** | **error** | **event**}

Example:

```
Router# debug acircuit event
```

Displays checkpoints, errors, and events that occur on the attachment circuits between the PE and CE routers.

Step 3 **debug ethernet service** {**all** | **api** | **error** | **evc** [*evc-id*] | **ha** | **instance** [**id** *id* | **interface** *type number* | **qos**] | **interface** *type number* | **microblock** | **oam-mgr**}

Example:

```
Router# debug ethernet service all
```

Enables debugging of Ethernet customer service instances.

Debugging mLACP on AToM Pseudowires

Use the **debug mpls l2transport vc** command for troubleshooting mLACP on AToM pseudowires.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug mpls l2transport vc** {**event** | **fsm** | **ldp** | **sss** | **status** {**event** | **fsm**}}

Example:

```
Router# debug mpls l2transport status event
```

Displays information about the status of AToM virtual circuits (VCs).

Debugging Cross-Connect Redundancy Manager and Session Setup

Use the following **debug**commands to troubleshoot cross-connect, redundancy manager, and session setup.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug sss error**

Example:

```
Router# debug sss error
```

Displays diagnostic information about errors that may occur during a subscriber service switch (SSS) call setup.

Step 3 **debug sss events**

Example:

```
Router# debug sss event
```

Displays diagnostic information about SSS call setup events.

Step 4 **debug xconnect** {**error** | **event**}

Example:

```
Router# debug xconnect event
```

Displays errors or events related to a cross-connect configuration.

Debugging VFI

Use the **debug vfi**command for troubleshooting a VFI.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug vfi** {**checkpoint** | **error** | **event** | **fsm** {**error** | **event**}}

Example:

```
Router# debug vfi checkpoint
```

Displays checkpoint information about a VFI.

Debugging the Segment Switching Manager (Switching Setup)

Use the **debug ssm**command for troubleshooting a segment switching manager (SSM).

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug ssm** {**cm errors** | **cm events** | **fhm errors** | **fhm events** | **sm errors** | **sm events** | **sm counters** | **xdr**}

Example:

```
Router# debug ssm cm events
```

Displays diagnostic information about the SSM for switched Layer 2 segments.

Debugging High Availability Features in mLACP

Use the following **debug**commands for troubleshooting High Availability features in mLACP.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **debug mpls l2transport checkpoint**

Example:

```
Router# debug mpls l2transport checkpoint
```

Enables the display of AToM events when AToM is configured for nonstop forwarding/stateful switchover (NSF/SSO) and Graceful Restart.

Step 3 **debug acircuit checkpoint**

Example:

```
Router# debug acircuit checkpoint
```

Enables the display of attachment circuit events when AToM is configured for NSF/SSO and Graceful Restart.

Step 4 **debug vfi checkpoint**

Example:

```
Router# debug vfi checkpoint
```

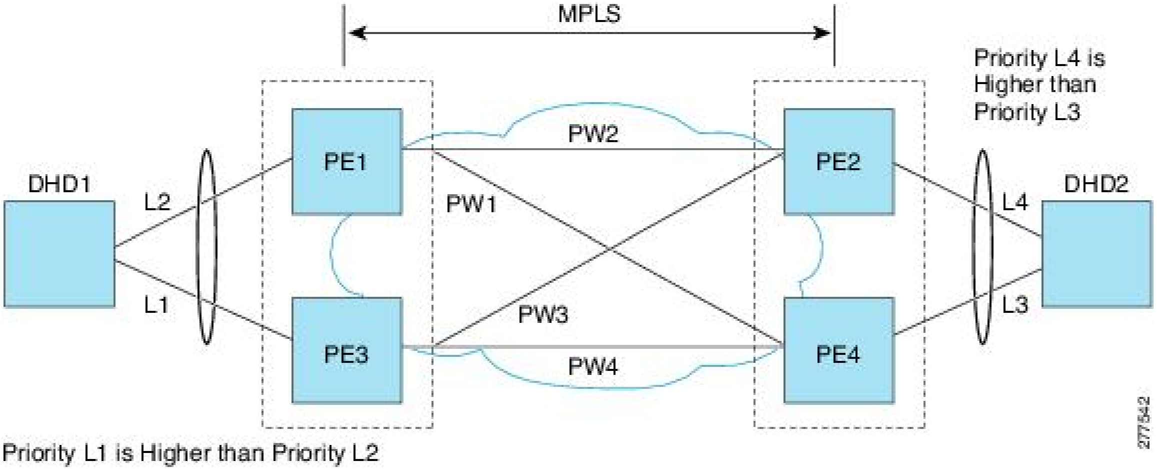
Enables the display of VFI events when AToM is configured for NSF/SSO and Graceful Restart.

Configuration Examples for mLACP

Example Configuring VPWS

Two sample configurations for VPWS follow: one example for an active PoA and the other for a standby PoA.

The figure below shows a sample topology for a VPWS configuration.


MPLS
Priority L4 is Higher than Priority L3
PE1
PE2
PE3
PE4
PW1
PW2
PW3
PW4
DHD1
DHD2
L1
L2
L3
L4
Priority L1 is Higher than Priority L2
277542

Active PoA for VPWS

The following VPWS sample configuration is for an active PoA:

```
mpls ldp graceful-restart
mpls label protocol ldp
!
redundancy
 mode sso
 interchassis group 1
  member ip 201.0.0.1
  backbone interface Ethernet0/2
  backbone interface Ethernet1/2
  backbone interface Ethernet1/3
  monitor peer bfd
  mlacp node-id 0
!
pseudowire-class mpls-dhd
 encapsulation mpls
 status peer topology dual-homed
!
interface Loopback0
 ip address 200.0.0.1 255.255.255.255
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
 mlacp interchassis group 1
 hold-queue 300 in
 service instance 1 ethernet
  encapsulation dot1q 100
  xconnect 210.0.0.1 10 pw-class mpls-dhd
   backup peer 211.0.0.1 10 pw-class mpls-dhd
!
interface Ethernet0/0
 no ip address
 channel-group 1 mode active
!
interface Ethernet1/3
 ip address 10.0.0.200 255.255.255.0
 mpls ip
 bfd interval 50 min_rx 150 multiplier 3
```

Standby PoA for VPWS

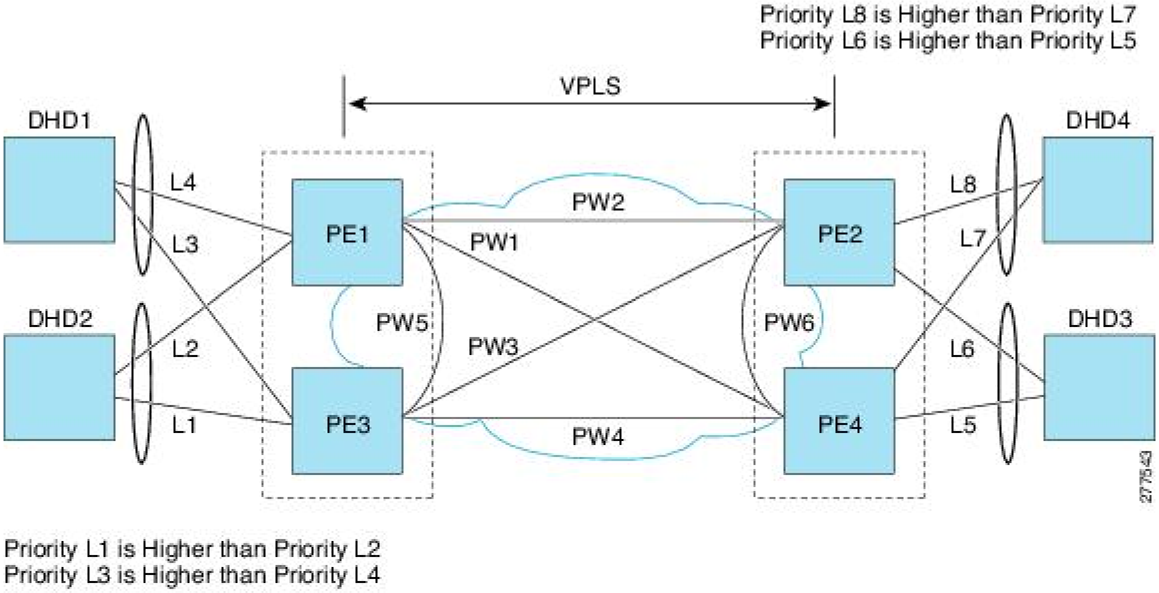
The following VPWS sample configuration is for a standby PoA:

```
mpls ldp graceful-restart
mpls label protocol ldp
mpls ldp graceful-restart
mpls label protocol ldp
!
Redundancy
 mode sso
 interchassis group 1
  member ip 200.0.0.1
  backbone interface Ethernet0/2
  backbone interface Ethernet1/2
  backbone interface Ethernet1/3
  monitor peer bfd
  mlacp node-id 1
!
pseudowire-class mpls-dhd
 encapsulation mpls
 status peer topology dual-homed
!
interface Loopback0
 ip address 201.0.0.1 255.255.255.255
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
 mlacp lag-priority 40000
 mlacp interchassis group 1
 hold-queue 300 in
 service instance 1 ethernet
  encapsulation dot1q 100
  xconnect 210.0.0.1 10 pw-class mpls-dhd
   backup peer 211.0.0.1 10 pw-class mpls-dhd
!
interface Ethernet1/0
 no ip address
 channel-group 1 mode active
!
interface Ethernet1/3
 ip address 10.0.0.201 255.255.255.0
 mpls ip
 bfd interval 50 min_rx 150 multiplier 3
```

Example Configuring VPLS

Two sample configurations for VPLS follow: one example for an active PoA and the other for a standby PoA.

The figure below shows a sample topology for a VPLS configuration.


Priority L8 is Higher than Priority L7
Priority L6 is Higher than Priority L5
VPLS
DHD1
DHD2
DHD3
DHD4
PE1
PE2
PE3
PE4
PW1
PW2
PW3
PW4
PW5
PW6
L1
L2
L3
L4
L5
L6
L7
L8
277543
Priority L1 is Higher than Priority L2
Priority L3 is Higher than Priority L4

Active PoA for VPLS

The following VPLS sample configuration is for an active PoA:

```
mpls ldp graceful-restart
mpls label protocol ldp
!
redundancy
 mode sso
 interchassis group 1
  member ip 201.0.0.1
  backbone interface Ethernet0/2
  monitor peer bfd
  mlacp node-id 0
!
l2 vfi VPLS_200 manual
 vpn id 10
 neighbor 210.0.0.1 encapsulation mpls
 neighbor 211.0.0.1 encapsulation mpls
 neighbor 201.0.0.1 encapsulation mpls
!
interface Loopback0
 ip address 200.0.0.1 255.255.255.255
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
 mlacp interchassis group 1
 service instance 1 ethernet
  encapsulation dot1q 100
  bridge-domain 200
!
interface Ethernet0/0
 no ip address
 channel-group 1 mode active
!
interface Ethernet1/3
```

```
 ip address 10.0.0.200 255.255.255.0
 mpls ip
 bfd interval 50 min_rx 150 multiplier 3
!
interface Vlan200
 no ip address
 xconnect vfi VPLS_200
```

Standby PoA for VPLS

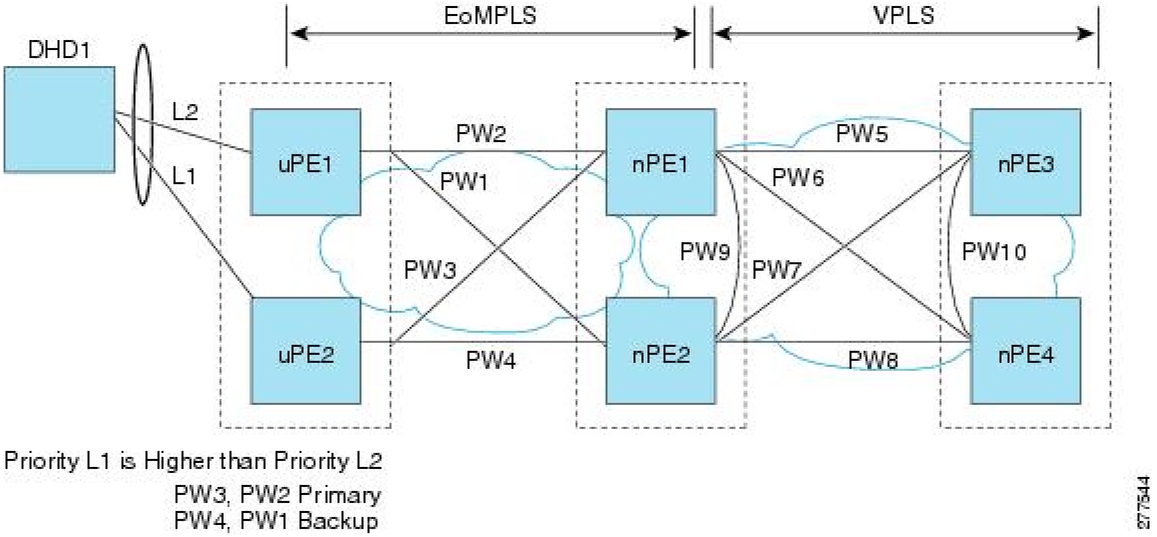
The following VPLS sample configuration is for a standby PoA:

```
mpls ldp graceful-restart
mpls label protocol ldp
!
redundancy
 interchassis group 1
  member ip 200.0.0.1
  backbone interface Ethernet0/2
  monitor peer bfd
  mlacp node-id 1
!
l2 vfi VPLS1 manual
 vpn id 10
 neighbor 210.0.0.1 encapsulation mpls
 neighbor 211.0.0.1 encapsulation mpls
 neighbor 200.0.0.1 encapsulation mpls
!
interface Loopback0
 ip address 201.0.0.1 255.255.255.255
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
 mlacp lag-priority 40000
 mlacp interchassis group 1
 service instance 1 ethernet
  encapsulation dot1q 100
  bridge-domain 200
!
interface Ethernet1/0
 no ip address
 channel-group 1 mode active
!
interface Ethernet1/3
 ip address 10.0.0.201 255.255.255.0
 mpls ip
 bfd interval 50 min_rx 150 multiplier 3
!
interface Vlan200
 no ip address
 xconnect vfi VPLS_200
```

Example Configuring H-VPLS

Two sample configurations for H-VPLS follow: one example for an active PoA and the other for a standby PoA.

The figure below shows a sample topology for a H-VPLS configuration.



Active PoA for H-VPLS

The following H-VPLS sample configuration is for an active PoA:

```
mpls ldp graceful-restart
mpls label protocol ldp
!
redundancy
 mode sso
 interchassis group 1
  member ip 201.0.0.1
  backbone interface Ethernet0/2
  backbone interface Ethernet1/2
  backbone interface Ethernet1/3
  monitor peer bfd
  mlacp node-id 0
!
pseudowire-class mpls-dhd
 encapsulation mpls
 status peer topology dual-homed
!
interface Loopback0
 ip address 200.0.0.1 255.255.255.255
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
 mlacp interchassis group 1
 hold-queue 300 in
 service instance 1 ethernet
  encapsulation dot1q 100
  xconnect 210.0.0.1 10 pw-class mpls-dhd
   backup peer 211.0.0.1 10 pw-class mpls-dhd
!
interface Ethernet0/0
 no ip address
 channel-group 1 mode active
!
interface Ethernet1/3
 ip address 10.0.0.200 255.255.255.0
 mpls ip
 bfd interval 50 min_rx 150 multiplier 3
```

Standby PoA for H-VPLS

The following H-VPLS sample configuration is for a standby PoA:

```
mpls ldp graceful-restart
mpls label protocol ldp
!
Redundancy
 mode sso
 interchassis group 1
  member ip 200.0.0.1
  backbone interface Ethernet0/2
  backbone interface Ethernet1/2
  backbone interface Ethernet1/3
  monitor peer bfd
  mlacp node-id 1
!
pseudowire-class mpls-dhd
 encapsulation mpls
 status peer topology dual-homed
!
interface Loopback0
 ip address 201.0.0.1 255.255.255.255
!
interface Port-channel1
 no ip address
 lacp fast-switchover
 lacp max-bundle 1
 mlacp lag-priority 40000
 mlacp interchassis group 1
 hold-queue 300 in
 service instance 1 ethernet
  encapsulation dot1q 100
  xconnect 210.0.0.1 10 pw-class mpls-dhd
   backup peer 211.0.0.1 10 pw-class mpls-dhd
!
interface Ethernet1/0
 no ip address
 channel-group 1 mode active
!
interface Ethernet1/3
 ip address 10.0.0.201 255.255.255.0
 mpls ip
 bfd interval 50 min_rx 150 multiplier 3
```

Example Verifying VPWS on an Active PoA

The following **show** commands can be used to display statistics and configuration parameters to verify the operation of the mLACP feature on an active PoA:

show lacp multichassis group

Use the **show lacp multichassis group** command to display the interchassis redundancy group value and the operational LACP parameters.

```
Router# show lacp multichassis group 100
```

```
Interchassis Redundancy Group 100
Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   0
System-Id: 200.000a.f331.2680
Peer Information:
State:        Up
Node-id:      7
System-Id:    2000.0014.6a8b.c680
ICCP Version: 0
State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        A/S     28000/32768        4/4             0/0
```

show lacp multichassis port-channel

Use the **show lacp multichassis port-channel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
Router# show lacp multichassis port-channel1
Interface Port-channel1
Local Configuration:
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0
Peer Configuration:
Interface: Port-channel1
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
                           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0
```

show mpls ldp iccp

Use the **show mpls ldp iccp** command to display the LDP session and ICCP state information.

```
Router# show mpls ldp iccp

ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1
```

show mpls l2transport

Use the **show mpls l2transport** command to display the local interface and session details, destination address, and status.

```
Router# show mpls l2transport vc 2
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
Po1            Eth VLAN 2                 172.2.2.2       2          UP
Po1            Eth VLAN 2                 172.4.4.4       2          STANDBY
```

show etherchannel summary

Use the **show etherchannel summary** command to display the status and identity of the mLACP member links.

```
Router# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
Number of channel-groups in use: 2
Number of aggregators:           2
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi2/9(P)    Gi2/20(P)   Gi2/31(P)
```

show etherchannel number port-channel

Use the **show etherchannel number port-channel** command to display the status and identity of the EtherChannel and and port channel.

```
Router# show etherchannel 51 port-c
```

```
                Port-channels in the group:
                ----------------------

Port-channel: Po51    (Primary Aggregator)

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

Age of the Port-channel   = 0d:02h:25m:23s
Logical slot/port   = 14/11          Number of ports = 2
HotStandBy port = null
Passive port list   = Gi9/15 Gi9/16
Port state          = Port-channel L3-Ag Ag-Inuse
Protocol            =   LACP
Fast-switchover     = enabled
Direct Load Swap    = disabled

Ports in the Port-channel:

Index   Load    Port      EC state        No of bits
------+------+--------+-----------------+-----------
  0     55     Gi9/15     mLACP-stdby   4
  1     AA     Gi9/16     mLACP-stdby   4

Time since last port bundled:    0d:01h:03m:39s    Gi9/16
Time since last port Un-bundled: 0d:01h:03m:40s    Gi9/16

Last applied Hash Distribution Algorithm: Fixed Channel-group Iedge Counts:
--------------------------:
Access ref count       : 0
Iedge session count    : 0
```

show lacp internal

Use the **show lacp internal**command to display the device, port, and member- link information.

```
Router# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi2/9     SA      bndl-act  28000         0x1       0x1     0x820A      0x3D
Gi2/20    SA      bndl-act  28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl-act  28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl-act  28000         0x1       0x1     0x8229      0x3D
Peer (MLACP-PE3) mLACP member links
Gi3/11    FA      hot-sby   32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      hot-sby   32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      hot-sby   32768         0x1       0x1     0xF321      0x7
Gi3/2     FA      hot-sby   32768         0x1       0x1     0xF303      0x7
```

Example Verifying VPWS on a Standby PoA

The following **show** commands can be used to display statistics and configuration parameters to verify the operation of the mLACP feature on a standby PoA:

show lacp multichassis group

Use the **show lacp multichassis group** command to display the LACP parameters, local configuration, status of the backbone uplink, peer information, node ID, channel, state, priority active, and inactive links.

```
Router# show lacp multichassis group 100
Interchassis Redundancy Group 100
Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   7
System-Id: 2000.0014.6a8b.c680
Peer Information:
State:        Up
Node-id:      0
System-Id:    200.000a.f331.2680
ICCP Version: 0
State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        S/A     32768/28000        4/4             0/0
```

show lacp multichassis portchannel

Use the **show lacp multichassis portchannel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
Router# show lacp multichassis port-channel1
Interface Port-channel1
Local Configuration:
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0
Peer Configuration:
Interface: Port-channel1
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
                           Bundled: 4
```

```
             Selected: 4
              Standby: 0
           Unselected: 0
```

show mpls ldp iccp

Use the **show mpls ldp iccp**command to display the LDP session and ICCP state information.

```
Router# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1
```

show mpls l2transport

Use the **show mpls l2transport** command to display the local interface and session details, destination address, and status.

```
Router# show mpls l2transport vc 2
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
Po1            Eth VLAN 2                 172.2.2.2       2          STANDBY
Po1            Eth VLAN 2                 172.4.4.4       2          STANDBY
```

show etherchannel summary

Use the **show etherchannel summary** command to display the status and identity of the mLACP member links.

```
Router# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
Number of channel-groups in use: 2
Number of aggregators:           2
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi3/2(P)    Gi3/11(P)   Gi3/21(P)
                                 Gi3/32(P)
```

show lacp internal

Use the **show lacp internal** command to display the device, port, and member-link information.

```
Router# show lacp 1 internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi3/2     FA      bndl-sby  32768         0x1       0x1     0xF303      0x7
Gi3/11    FA      bndl-sby  32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      bndl-sby  32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      bndl-sby  32768         0x1       0x1     0xF321      0x7
Peer (MLACP-PE1) mLACP member links
Gi2/20    SA      bndl      28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl      28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl      28000         0x1       0x1     0x8229      0x3D
Gi2/9     SA      bndl      28000         0x1       0x1     0x820A      0x3D
```

Example Verifying VPLS on an Active PoA

The following **show** commands can be used to display statistics and configuration parameters to verify the operation of the mLACP feature on an active PoA:

show lacp multichassis group

Use the **show lacp multichassis group** command to display the LACP parameters, local configuration, status of the backbone uplink, peer information, node ID, channel, state, priority active, and inactive links.

```
Router# show lacp multichassis group 100
Interchassis Redundancy Group 100
Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   0
System-Id: 200.000a.f331.2680
Peer Information:
State:        Up
Node-id:      7
System-Id:    2000.0014.6a8b.c680
ICCP Version: 0
State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        A/S     28000/32768        4/4            0/0
```

show lacp multichassis port-channel

Use the **show lacp multichassis port-channel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
Router# show lacp multichassis port-channel1
Interface Port-channel1
Local Configuration:
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0
Peer Configuration:
Interface: Port-channel1
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
                           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0
```

show mpls ldp iccp

Use the **show mpls ldp iccp** command to display the LDP session and ICCP state information.

```
Router# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1
```

show mpls l2transport

Use the **show mpls l2transport** command to display the local interface and session details, destination address, and the status.

```
Router# show mpls l2transport vc 4000
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
VFI VPLS       VFI                        172.2.2.2       4000       UP
VFI VPLS       VFI                        172.4.4.4   4000       UP
```

show etherchannel summary

Use the **show etherchannel summary** command to display the status and identity of the mLACP member links.

```
Router# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
Number of channel-groups in use: 2
Number of aggregators:           2
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi2/9(P)    Gi2/20(P)   Gi2/31(P)
                                 Gi2/40(P)
```

show lacp internal

Use the **show lacp internal** command to display the device, port, and member-link information.

```
Router# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi2/9     SA      bndl-act  28000         0x1       0x1     0x820A      0x3D
Gi2/20    SA      bndl-act  28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl-act  28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl-act  28000         0x1       0x1     0x8229      0x3D
Peer (MLACP-PE3) mLACP member links
Gi3/11    FA      hot-sby   32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      hot-sby   32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      hot-sby   32768         0x1       0x1     0xF321      0x7
Gi3/2     FA      hot-sby   32768         0x1       0x1     0xF303      0x7
```

Example Verifying VPLS on a Standby PoA

The **show** commands in this section can be used to display statistics and configuration parameters to verify the operation of the mLACP feature:

show lacp multichassis group

Use the **show lacp multichassis group** *interchassis group number* command to display the LACP parameters, local configuration, status of the backbone uplink, peer information, node ID, channel, state, priority, active, and inactive links.

```
Router# show lacp multichassis group 100
Interchassis Redundancy Group 100
Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   7
System-Id: 2000.0014.6a8b.c680
Peer Information:
State:        Up
Node-id:      0
System-Id:    200.000a.f331.2680
ICCP Version: 0
State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        S/A     32768/28000        4/4             0/0
```

show lacp multichassis portchannel

Use the **show lacp multichassis portchannel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
Router# show lacp multichassis port-channel1
Interface Port-channel1
Local Configuration:
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0
Peer Configuration:
Interface: Port-channel1
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
```

```
                           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0
```

show mpls ldp iccp

Use the **show mpls ldp iccp** command to display the LDP session and ICCP state information.

```
Router# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1
```

show mpls l2transport

Use the **show mpls l2transport** command to display the local interface and session details, destination address, and status.

```
Router# show mpls l2transport vc 4000
Local intf     Local circuit          Dest address    VC ID      Status
-------------  -------------------- --------------- ---------- ----------
VFI VPLS       VFI                     172.2.2.2       4000       UP
VFI VPLS       VFI                     172.4.4.4       4000       UP
```

showetherchannelsummary

Use the **show etherchannel summary** command to display the status and identity of the mLACP member links.

```
Router# show etherchannel summary

Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
Number of channel-groups in use: 2
Number of aggregators:           2
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
```

```
1       Po1(RU)          LACP      Gi3/2(P)    Gi3/11(P)   Gi3/21(P)
                                   Gi3/32(P)
```

show lacp internal

Use the **show lacp internal** command to display the device, port, and member- link information.

```
Router# show lacp 1 internal

Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode
Channel group 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi3/2     FA      bndl-sby  32768         0x1       0x1     0xF303      0x7
Gi3/11    FA      bndl-sby  32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      bndl-sby  32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      bndl-sby  32768         0x1       0x1     0xF321      0x7
Peer (MLACP-PE1) mLACP member links
Gi2/20    SA      bndl      28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl      28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl      28000         0x1       0x1     0x8229      0x3D
Gi2/9     SA      bndl      28000         0x1       0x1     0x820A      0x3D
```

Feature Information for mLACP

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

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

Table 27: Feature Information for mLACP

Feature Name	Releases	Feature Information
Multichassis LACP (mLACP)		Cisco's mLACP feature addresses the need for interchassis redundancy mechanisms when a carrier wants to dual home a device to two upstream PoAs for redundancy. The mLACP feature enhances the 802.3ad LACP implementation to meet this requirement. The following commands were introduced or modified: **backbone interface**, **debug acircuit checkpoint**, **debug lacp**, **ethernet mac-flush mirp notification**, **interchassis group**, **lacp failover**, **lacp max-bundle**, **lacp min-bundle**, **member ip**, **mlacp interchassis group**, **mlacp lag-priority**, **mlacp node-id**, **mlacp system-mac**, **mlacp system-priority**, **monitor peer bfd**, **redundancy**, **show ethernet service instance interface port-channel**, **show ethernet service instance id mac-tunnel**, **show lacp**, **status decoupled**, **status peer topology dual-homed**.

Glossary

active attachment circuit—The link that is actively forwarding traffic between the DHD and the active PoA.

active PW—The pseudowire that is forwarding traffic on the active PoA.

BD—bridge domain.

BFD—bidirectional forwarding detection.

DHD—dual-homed device. A node that is connected to two switches over a multichassis link aggregation group for the purpose of redundancy.

DHN—dual-homed network. A network that is connected to two switches to provide redundancy.

H-VPLS—Hierarchical Virtual Private LAN Service.

ICC—Interchassis Communication Channel.

ICCP—Interchassis Communication Protocol.

ICPM—Interchassis Protocol Manager.

ICRM—Interchassis Redundancy Manager.

LACP—Link Aggregation Control Protocol.

LAG—link aggregation group.

LDP—Link Distribution Protocol.

MCEC—Multichassis EtherChannel.

mLACP—Multichassis LACP.

PoA—point of attachment. One of a pair of switches running multichassis link aggregation group with a DHD.

PW-RED—pseudowire redundancy.

standby attachment circuit—The link that is in standby mode between the DHD and the standby PoA.

standby PW—The pseudowire that is in standby mode on either an active or a standby PoA.

uPE—user-facing Provider Edge.

VPLS—Virtual Private LAN Service.

VPWS—Virtual Private Wire Service.

CHAPTER 18

ICCP Multichassis VLAN Redundancy

Carrier Ethernet network high availability can be achieved by employing intra- and inter-chassis redundancy mechanisms. The Multichassis Link Aggregation Control Protocol (mLACP) solution addresses the latter, where a carrier wants dual-homed device (DHD) to two upstream points of attachment (PoA) for redundancy. Some carriers do not run loop prevention control protocols in their access networks, so an alternate redundancy scheme is necessary.

The implementation of mLACP supports DHD with an active/standby topology. Interchassis Communication Protocol (ICCP) Multichassis VLAN Redundancy, also known as Pseudo mLACP, provides a flexible dual-homing redundancy mechanism. It uses similar principles as mLACP. The Pseudo mLACP solution extends the mLACP functionality to support active/active PoAs deployments. This enables flexibility in network planning and efficient resource utilization.

Pseudo mLACP has the following advantages over mLACP:

- Pseudo mLACP supports per-VLAN active/active redundancy without any load-balancing requirements on the CE.
- Pseudo mLACP is independent of the access redundancy mechanism; therefore, it provides a network-based redundancy solution. It allows maximum flexibility for the Provider Edge (PE)-Customer Edge (CE) interoperability in terms of dual-homing redundancy and recovery.

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.

Prerequisites for ICCP Multichassis VLAN Redundancy

- mLACP support is required for Pseudo mLACP.

Restrictions for ICCP Multichassis VLAN Redundancy

- Max bundle should not be configured on a Pseudo mLACP enabled port channel.
- Pseudo mLACP does not work with most of the Layer 2 control protocols or Spanning Tree Protocol (STP) including Multiple Spanning Tree Protocol (MSTP) or VLAN Trunking Protocol (VTP).
- When a service instance is configured under a Pseudo mLACP port channel, all the outer tag VLANs of a service instance must be a part of either a primary VLAN list or a secondary VLAN list.
- Outer VLANs of one service instance cannot be mixed with the primary and secondary VLAN list on a Pseudo mLACP port channel.
- Brute-Force mode configuration is not supported.
- VLAN force-switchover configuration is applicable only for nonrevertive mode.
- The DHD nodes must support the LACP functionality.
- The DHD nodes must support MVRP MAC flush functionality in Pseudo mLACP topology.

Information About ICCP Multichassis VLAN Redundancy

Pseudo mLACP Multihoming Redundancy

The provider edge (PE) ports are configured in such a way that they act as if connected to a virtual device over a Multichassis link aggregation group (MC-LAG) with mLACP. Points of Attachment (PoAs) can be placed in active/active mode with manual VLAN load balancing. DHD ports are configured into two individual port channels that are physically connected to each of the PoAs. Interchassis Communication Protocol (ICCP), with new extensions is used for interchassis communication to control the failover process. Multiple VLAN Registration Protocol (MVRP) lite is used for active VLAN notification and MAC flushing toward the access side. For MAC flushing notification toward the core, MVRP lite, Multiple I-SID Registration Protocol (MIRP) lite, or LDP MAC withdraw can be used.

Pseudo mLACP provides:

- The active/active mode redundancy of two PoAs in a redundancy group. This provides higher bandwidth utilization than mLACP and other active/standby link-level schemes. Pseudo mLACP eliminates the required wasted link bandwidth on the standby PoA.
- Flexible access network topologies, that is, access network dual-homing and access device dual-homing.

- Service provider control over the provisioning, role assignment, failover, and load sharing between PoAs.
- PE node redundancy for Virtual Private Wire Service (VPWS), Virtual Private LAN Service (VPLS), and native Ethernet aggregation.

The DHD is configured with two different port channels that are connected to a single multichassis LAG (mLAG) on the PoA side. The LACP module on the PoAs receives two different port keys from the two different port channels on the DHD. The mLAG on the PoA ignores the port keys from the DHD's LACP PDUs to form a single bundle on the PoA side.

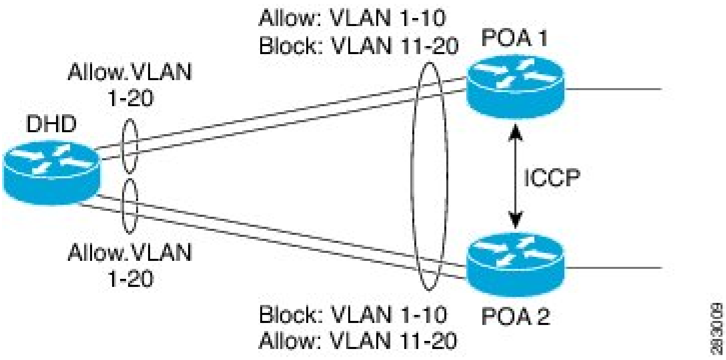
The mLACP module provides failover and recovery notifications to Pseudo mLACP. Reversion delay is processed by the mLACP module. mLACP provides a CLI interface for Pseudo mLACP VLANs and mode configuration. mLACP supports VLAN-based active/active redundancy, in addition to PoA-level active/standby redundancy. VLAN-based active/active redundancy allows you to bundle links on both the PoAs based on the Pseudo mLACP configuration. Pseudo mLACP and mLACP port-channels can be configured together on the same pair of PoAs, and both can use the same redundancy group.

After failover, the new active PoA activates the standby VLAN list on the port-channel. However, to receive traffic on the newly active VLAN's DHD, networks must flush their MAC address table and learn the new MAC address of the new PoA port channel interface. The existing MVRP lite support is used for DHD-side MAC flushing.

Pseudo mLACP Active/Active Support

Pseudo mLACP supports active/active redundancy without the restriction of symmetric VLAN-based load sharing in both the Provider Edge (PE) and the Customer Edge (CE).

Figure 5: Active/Active Support



Pseudo mLACP provides VLAN-based redundancy by allowing you to specify one primary interface and one secondary interface or a PoA pair for each member VLAN. The configuration determines the PoA that will be initially active for a VLAN, by using the primary and secondary VLAN lists under the Pseudo mLACP interface. Only the active PoA will forward frames for the respective VLANs. The standby PoA will be in the blocking mode (bidirectional), dropping all the frames received on the standby VLANs. The failover will occur for all the VLANs in the active/standby list and not on a per-VLAN basis. Pseudo mLACP provides per-port-channel VLAN load balancing. You can statistically configure the primary and secondary VLAN list on each of the PoAs.

The DHD nodes are configured such that each of their uplinks to a PoA operates as an individual port channel. Each interface must be configured to forward all local VLANs on all uplinks belonging to the mLAG.

The data-path forwarding scheme causes the DHD to automatically learn which PoA or interface is active for a given VLAN. This learning occurs at an individual destination MAC address level.

Failure Recovery

Pseudo mLACP uses revertive behavior (which is the default behavior) after the failure recovery to support the active/active model. You can configure a nonrevertive mode.

Reversion occurs the same way that the original failover occurs. The reversion must be initiated by the new active PoA for the given VLANs, by signaling that the PoA is relinquishing its active role for the VLAN. This is done through an ICCP Pseudo mLACP port-state TLV, which indicates that it is no longer in the active mode for the affected VLANs. Upon a TLV receipt, the recovering PoA unblocks the affected VLANs, and triggers MAC flushes toward both the access side and the core side).

mLACP reversion delay applies for Pseudo mLACP operations. However, reversion occurs only for failed-over VLANs. The forced failover mechanism based on dynamic port-priority change cannot be used for Pseudo mLACP because all the member links will remain in the bundle state. Use the **mlacp reversion-delay** command to configure the mLACP reversion timer. Use the **mlacp load-balance force-switchover portchannel** command to configure forced VLAN switchover.

Pseudo mLACP Failover Operations

The Pseudo mLACP forces a PoA failover to the standby PoA when one of the following failures occurs:

Note mLACP failover will not be triggered if Pseudo mLACP is not configured correctly. If the mLACP failover occurs before the peer PoA is configured with Pseudo mLACP, the failover will occur as long as the peer PoA meets the mLACP failover requirements.

- Access side link or port failure—This failure is triggered by a min-link failure. On receiving a min-link failure, all the active VLANs on the port-channel failover to the other PoA. This failover is initiated by sending a Pseudo mLACP PORT-STATE TLV message, indicating that the port state is DOWN.
- Node failure—The surviving PoA's Pseudo mLACP receives notification of node failure and initiates failover of all VLANs that were in standby mode on all shared mLAGs. After recovery, both POAs synchronize again.
- PoA uplink failure—The failing PoA signals the peer about the core isolation using the Pseudo mLACP PORT-STATE TLV, indicating that the PoA is isolated. It places all the VLANs in the blocking mode.

How to Configure ICCP Multichassis VLAN Redundancy

Configuring a Port Channel for Pseudo mLACP

Perform this task to configure a port channel for Pseudo mLACP.

Before you begin

Note The redundancy group should be configured. Redundancy group configuration for Pseudo mLACP is the same as for mLACP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *number*
4. **mlacp interchassis group** *group-id*
5. **mlacp mode active-active**
6. **mlacp load-balance primary vlan** *vlan-id*
7. **mlacp load-balance secondary vlan** *vlan-id*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *number* **Example:** `Router(config)# interface port-channel 1`	Configures the port channel and enters interface configuration mode.
Step 4	**mlacp interchassis group** *group-id* **Example:** `Router(config-if)# mlacp interchassis group 1`	Specifies that the port channel is an mLACP port-channel .
Step 5	**mlacp mode active-active** **Example:** `Router(config-if)# mlacp mode active-active`	Enables pseudo mLACP operations on the PoA and allows the PoA to form an LACP bundle even if the partner receives an LACP PDU from two different port channels on a dual-homed network (DHN) or dual-homed device (DHD).
Step 6	**mlacp load-balance primary vlan** *vlan-id* **Example:** `Router(config-if)# mlacp load-balance primary vlan 10,20`	Configures the list of primary VLANs that will be active and inactive on the given PoA.

	Command or Action	Purpose
Step 7	**mlacp load-balance secondary vlan** *vlan-id* **Example:** `Router(config-if)# mlacp load-balance secondary vlan 30,100`	Configures the list of secondary VLANs that will be active and inactive on the given PoA.

Step 8	**end**
	Example:
	`Router(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for ICCP Multichassis VLAN Redundancy

Example: Port Channel Configuration for Pseudo mLACP

The following example shows how to configure the port channel on the active and standby PoA for Pseudo mLACP.

Active PoA-POA1

```
Router# configure terminal
Router(config)# interface port-channel1
Router(config-if)# mlacp interchassis group 1
Router(config-if)# mlacp mode active-active
Router(config-if)# mlacp load-balance primary vlan 10,20
Router(config-if)# mlacp load-balance secondary vlan 30,100
Router(config-if)# end
```

Standby PoA-POA2

```
Router# configure terminal
Router(config)# interface port-channel1
Router(config-if)# mlacp interchassis group 1
Router(config-if)# mlacp mode active-active
Router(config-if)# mlacp load-balance primary vlan 30,100
Router(config-if)# mlacp load-balance secondary vlan 10,20
Router(config-if)# end
```

Feature Information for ICCP Multichassis VLAN Redundancy

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

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

Table 28: Feature Information for ICCP Multiichassis VLAN Redundancy

Feature Name	Releases	Feature Information
ICCP Multichassis VLAN Redundancy		Pseudo mLACP provides a flexible dual-homing redundancy mechanism. It uses similar principles as mLACP, but without the implementation of LACP between the PEs and CEs. The PE ports are configured in such a way that they act as if connected to a virtual device over an MC-LAG with mLACP. Ports can be placed in active/active mode with manual VLAN load balancing. The following commands were introduced or modified: **debug lacp**, **debug mvrp**, **mlacp load-balance**, **mlacp load-balance force-switchover**, **mlacp mode active-active**, **mlacp reversion-delay**, **show lacp**.

Glossary

active attachment circuit—The link that is actively forwarding traffic between the DHD and the active PoA.

active PW—The pseudowire that is forwarding traffic on the active PoA.

BD—bridge domain.

BFD—bidirectional forwarding detection.

DHD—dual-homed device. A node that is connected to two switches over a multichassis link aggregation group for the purpose of redundancy.

DHN—dual-homed network. A network that is connected to two switches to provide redundancy.

H-VPLS—Hierarchical Virtual Private LAN Service.

ICC—Interchassis Communication Channel.

ICCP—Interchassis Communication Protocol.

ICPM—Interchassis Protocol Manager.

ICRM—Interchassis Redundancy Manager.

LACP—Link Aggregation Control Protocol.

LAG—link aggregation group.

LDP—Link Distribution Protocol.

MCEC—Multichassis EtherChannel.

mLACP—Multichassis LACP.

PoA—point of attachment. One of a pair of switches running multichassis link aggregation group with a DHD.

PW-RED—pseudowire redundancy.

standby attachment circuit—The link that is in standby mode between the DHD and the standby PoA.

standby PW—The pseudowire that is in standby mode on either an active or a standby PoA.

uPE—user-facing Provider Edge.

VPLS—Virtual Private LAN Service.

VPWS—Virtual Private Wire Service.

CHAPTER 19

MC-LAG TCN Interworking

Multiple VLAN Registration Protocol (MVRP) is used for MAC Flushing during the Pseudowire (PW) redundancy process. However, not all Dual Homed Device (DHD) switches support MVRP for MAC flushing. MC-LAG TCN Interworking feature enables using the Multiple Spanning Tree Protocol with Topology Change Notification (MSTP TCN) scheme for MAC flushing towards the access network.

Prerequisites for MC-LAG TCN Interworking

- Ethernet Flow Points (EFPs) towards the core network as well as the access network must support the MSTP instance (creation and deletion) for sending and receiving Bridge Protocol Data Units (BPDUs).
- DHD access node(s) must support MSTP TCN.
- To enable the MAC mode for multichassis LACP (mLACP) or Pseudo mLACP (P- mLACP), mLACP sub-block must be created first.
- MSTP TCN enabled port channel interface must be compliant with High Availability (HA) synchronization (between HA Active and HA Hot Standby).

Restrictions for MC-LAG TCN Interworking

- P-mLACP mode needs to be configured before enabling MSTP TCN.
- The port channel configuration on both Point of Attachments (PoAs) must be same, including EFP IDs.
- Port channel members need not be same on PoAs.
- Each PoA may be connected to the DHD with a different number of links for the Link Aggregation Group (LAG) (and hence configured with a different value for the max-links value) variable.

- Virtual Private Wire Service (VPWS) and Virtual Private LAN Service (VPLS) VC state (Active/Standby) are based on the Active VLAN list configuration on a PoA at any given time.

Information About MC-LAG TCN Interworking

MC-LAG TCN Interworking

Multiple Spanning Tree Protocol (MSTP) is an extension of the original STP specification. It is an IETF standard stack with a completed state machine (SM) for processing root path costs, topology change notification of the port or VLAN, and so on. MSTP uses Bridge Protocol Data Units (BPDU) to exchange information such as bridge IDs or root path costs. There are two types of BPDU in the MST stack.

- Configuration BPDU (CBPDU)
- Topology Change Notification BPDU (TCN BPDU)

Within the MST, BPDUs are exchanged regularly and enable devices to keep track of network changes and to start and stop forwarding at ports as required. MC-LAG TCN Interworking feature uses TCN BPDU to announce the changes in the network topology to access side DHD, requesting for MAC flushing. The DHD processes the MST TCN message and updates the forwarding table with appropriate outgoing interface for each destination MAC address.

MAC flushing is triggered during the following conditions:

- Pseudowire (PW) redundancy has taken place for switchover between VLANs or POAs.
- VLAN configuration has been changed by the administrator.

MSTP Topology Change Notification scheme can be configured per port-channel basis for MAC Flushing. MVRP Lite is used for MAC flushing during redundancy switchover as a default scheme.

How to Configure MC-LAG TCN Interworking

Enabling MSTP TCN Sequence

Before you begin

Note Enable P-mLACP feature before enabling MSTP TCN sequence.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *number*
4. **mlacp interchassis group** *group-id*

5. **mlacp mode active-active**
6. **mlacp mac mstp-tcn**
7. **mlacp load-balance primary vlan** *vlan-id*
8. **mlacp load-balance secondary vlan** *vlan-id*
9. **end**
10. Perform the same steps on standby POA.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *number* **Example:** `Device(config)# interface port-channel 1`	Configures the port channel and enters interface configuration mode.
Step 4	**mlacp interchassis group** *group-id* **Example:** `Device(config-if)# mlacp interchassis group 1`	Specifies that the port channel is an mLACP port channel.
Step 5	**mlacp mode active-active** **Example:** `Device(config-if)# mlacp mode active-active`	Enables P-mLACP operations on a PoA and allows the PoA to form an LACP bundle even if the peer receives an LACP protocol data unit (PDU) from two different port channels on a dual-homed network (DHN) or DHD.
Step 6	**mlacp mac mstp-tcn** **Example:** `Device(config-if)# mlacp mac mstp-tcn`	Enables MAC mode on port channel base.
Step 7	**mlacp load-balance primary vlan** *vlan-id* **Example:** `Device(config-if)# mlacp load-balance primary vlan 10,20`	Configures a list of primary VLANs that will be active on a given PoA.
Step 8	**mlacp load-balance secondary vlan** *vlan-id* **Example:** `Device(config-if)# mlacp load-balance secondary vlan 30,100`	Configures a list of secondary VLANs that will be standby on a given PoA.
Step 9	**end** **Example:**	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	
Step 10	Perform the same steps on standby POA.	—

Enabling MST for VLANs

SUMMARY STEPS

1. **configure terminal**
2. **spanning-tree mode mst**
3. **spanning-tree extend system-id**
4. **spanning-tree mst configuration**
5. **name** *name*
6. **revision** *version*
7. **instance** *instance-id* **vlan** *vlan-range*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**spanning-tree mode mst** **Example:** `Device(config)# spanning-tree mode mst`	Enables MST on the device.
Step 3	**spanning-tree extend system-id** **Example:** `Device(config)# spanning-tree extend system-id`	Enables the extended-system ID.
Step 4	**spanning-tree mst configuration** **Example:** `Device(config)# spanning-tree mst configuration`	Enters MST configuration submode on the system.
Step 5	**name** *name* **Example:** `Device(config-mst)# name test`	Specifies the name of an MST region
Step 6	**revision** *version* **Example:** `Device(config-mst)# revision 1`	Specifies the revision number for the MST configuration

	Command or Action	Purpose

Step 7	**instance** *instance-id* **vlan** *vlan-range*
	Example:
	`Device(config-mst)# instance 1 vlan 1-63`
	`Device(config-mst)# instance 1 vlan 20, 40`
		Maps VLANs to an MST instance.
		• *instance-id*—Range is 0 to 4094.
		• *vlan-range*—Range is 1 to 4094.
		To specify a VLAN range, use a hyphen; for example, **instance 1 vlan 1-63** maps VLANs 1 through 63 to MST instance 1.
		To specify a VLAN series, use a comma; for example, **instance 1 vlan 20, 40** maps VLANs 20 and 40 to MST instance 1.

Step 8	**exit** **Example:** `Device(config-mst)# exit`	Exits MST configuration mode and returns to global configuration mode.

Verifying MC-LAG TCN Interworking

All steps are optional and can be performed in any order.

SUMMARY STEPS

1. **enable**
2. **show ethernet service interface** [*type number*] [**detail**]
3. **show spanning-tree detail**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables the privileged EXEC mode. Enter your password if prompted.

Step 2 **show ethernet service interface** [*type number*] [**detail**]

Example:

```
Device(config)# show ethernet service interface port 1 detail

 Interface: Port-channel1, Type: UNI
 ID:
 EVC Distribution State: Ready
 EVC Map Type: Bundling-Multiplexing
 Bridge-domains:
 Associated Service Instances:
    Service-Instance-ID CE-VLAN
    20
    40
 L2protocol pass

mLACP state: Active
```

Displays the information about mLACP enabled Ethernet interface port.

Step 3 **show spanning-tree detail**

Example:

```
Device# show spanning-tree detail

MST0 is executing the mstp compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, sysid 0, address f866.f2eb.7ebb
  Configured hello time 2, max age 20, forward delay 15, transmit hold-count 6
  Current root has priority 32768, address 2834.a252.7380
  Root port is 14 (Port-channel1), cost of root path is 0
  Topology change flag not set, detected flag not set
  Number of topology changes 2 last change occurred 00:15:24 ago
          from Port-channel1
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0

Port 14 (Port-channel1) of MST0 is root forwarding
   Port path cost 20000, Port priority 128, Port Identifier 128.14.
   Designated root has priority 32768, address 2834.a252.7380
   Designated bridge has priority 32768, address 2834.a252.7380
   Designated port id is 128.456, designated path cost 0
   Timers: message age 4, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default, Internal
   BPDU: sent 8, received 774

 MST1 is executing the mstp compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, sysid 1, address f866.f2eb.7ebb
  Configured hello time 2, max age 20, forward delay 15, transmit hold-count 6
  Current root has priority 32769, address 2834.a252.7380
  Root port is 14 (Port-channel1), cost of root path is 20000
  Topology change flag not set, detected flag not set
  Number of topology changes 3 last change occurred 00:12:04 ago
          from Port-channel1
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0

 Port 14 (Port-channel1) of MST1 is root forwarding
   Port path cost 20000, Port priority 128, Port Identifier 128.14.
   Designated root has priority 32769, address 2834.a252.7380
   Designated bridge has priority 32769, address 2834.a252.7380
   Designated port id is 128.456, designated path cost 0
   Timers: message age 5, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default, Internal
   BPDU: sent 8, received 775
```

Displays the STP details including TCN information.

Configuration Examples for MC-LAG TCN Interworking

Example: Enabling MSTP TCN Sequence

The following example shows how to enable the MSTP TCN sequence.

Active PoA-POA1

```
Device# configure terminal
Device(config)# interface port-channel1
Device(config-if)# mlacp interchassis group 1
Device(config-if)# mlacp mode active-active
Device(config-if)# mlacp mac mstp-tcn
Device(config-if)# mlacp load-balance primary vlan 10,20
Device(config-if)# mlacp load-balance secondary vlan 30,100
Device(config-if)# end
```

Standby PoA-POA2

```
Device# configure terminal
Device(config)# interface port-channel1
Device(config-if)# mlacp interchassis group 1
Device(config-if)# mlacp mode active-active
Device(config-if)# mlacp mac mstp-tcn
Device(config-if)# mlacp load-balance primary vlan 30,100
Device(config-if)# mlacp load-balance secondary vlan 10,20
Device(config-if)# end
```

Example: Enabling MST for VLANs

The following example shows the STP configuration for VLANs 20 and 40.

```
Device# configure terminal
Device(config)# spanning-tree mode mst
Device(config)# spanning-tree extend system-id
Device(config)# spanning-tree mst configuration
Device(config-mst)# name test
Device(config-mst)# revision 1
Device(config-mst)# instance 1 vlan 20, 40
```

Example: Configuring Redundancy and P-mLACP on Active POA

The following example shows how to configure redundancy and P-mLACP on an active POA.

```
redundancy
 mode sso
  interchassis group 4294967295
```

```
   monitor peer bfd
   member ip 88.1.1.2
   backbone interface GigabitEthernet0/0/2
   backbone interface GigabitEthernet0/0/1
   mlacp system-mac 0001.0001.0001
   mlacp system-priority 100
   mlacp node-id 1
 !
 !
interface Port-channel1
no ip address
no negotiation auto
mlacp interchassis group 4294967295
mlacp mode active-active
mlacp mac mstp-tcn
mlacp load-balance primary vlan 40
mlacp load-balance secondary vlan 20
service instance 20 ethernet
  encapsulation dot1q 20
  rewrite ingress tag pop 1 symmetric
  xconnect 88.1.1.3 20 encapsulation mpls pw-class poa
backup peer 88.1.1.4 20 pw-class poa
!
service instance 40 ethernet
  encapsulation dot1q 40
  rewrite ingress tag pop 1 symmetric
  xconnect 88.1.1.3 40 encapsulation mpls pw-class poa
   backup peer 88.1.1.4 40 pw-class poa
!
interface Port-channel10
 description to-DHD
 no ip address
 mlacp interchassis group 100
 mlacp mode active-active
mlacp mac mstp-tcn
 mlacp load-balance primary vlan 100-109
 mlacp load-balance secondary vlan 110-120
  service instance 10 ethernet
  encapsulation dot1q 100
  rewrite ingress tag pop 1 symmetric
  xconnect 3.3.3.3 90 encapsulation mpls
!
 service instance 11 ethernet evc11_bd_201
  encapsulation dot1q 101
  rewrite ingress tag pop 1 symmetric
  bridge-domain 201
!
 service instance 12 ethernet
  encapsulation dot1q 102
  rewrite ingress tag pop 1 symmetric
  bridge-domain 202 split-horizon
 !
 service instance 20 ethernet
  encapsulation dot1q 110
  rewrite ingress tag pop 1 symmetric
  xconnect 3.3.3.3 91 encapsulation mpls
!
service instance 21 ethernet
  encapsulation dot1q 111
  rewrite ingress tag pop 1 symmetric
  bridge-domain 211
 !
 service instance 22 ethernet
  encapsulation dot1q 112
  rewrite ingress tag pop 1 symmetric
  bridge-domain 212 split-horizon
 !
```

Example: Configuring Redundancy and P-mLACP on Standby POA

The following example shows how to configure redundancy and P-mLACP on a standby POA.

```
redundancy
 mode sso
 interchassis group 100
  monitor peer bfd
  member ip 1.1.1.1
  backbone interface GigabitEthernet8/0/10
  mlacp system-priority 100
  mlacp node-id 2



interface Port-channel1
 no ip address
 no negotiation auto
 mlacp interchassis group 4294967295
 mlacp mode active-active
 mlacp mac mstp-tcn
 mlacp load-balance primary vlan 20
 mlacp load-balance secondary vlan 40
 service instance 40  ethernet
   encapsulation dot1q 40
 rewrite ingress tag pop 1 symmetric
   xconnect 88.1.1.3 20 encapsulation mpls pw-class poa
    backup peer 88.1.1.4 20 pw-class poa
 !
 service instance 20 ethernet
   encapsulation dot1q 20
   rewrite ingress tag pop 1 symmetric
   xconnect 88.1.1.3 20 encapsulation mpls pw-class poa
    backup peer 88.1.1.4 20 pw-class poa
 !
interface Port-channel10
 description to-DHD
 no ip address
 mlacp interchassis group 100
 mlacp mode active-active
mlacp mac mstp-tcn
 mlacp load-balance primary vlan 110-120
 mlacp load-balance secondary vlan 100-109
  service instance 10 ethernet
  encapsulation dot1q 100
  rewrite ingress tag pop 1 symmetric
  xconnect 3.3.3.3 90 encapsulation mpls
   !
 service instance 11 ethernet
  encapsulation dot1q 101
  rewrite ingress tag pop 1 symmetric
  bridge-domain 201
 !
 service instance 12 ethernet
  encapsulation dot1q 102
  rewrite ingress tag pop 1 symmetric
  bridge-domain 202 split-horizon
 !
 service instance 20 ethernet
  encapsulation dot1q 110
  rewrite ingress tag pop 1 symmetric
  xconnect 3.3.3.3 91 encapsulation mpls
   !
service instance 21 ethernet
  encapsulation dot1q 111
  rewrite ingress tag pop 1 symmetric
  bridge-domain 211
 !
 service instance 22 ethernet
  encapsulation dot1q 112
  rewrite ingress tag pop 1 symmetric
  bridge-domain 212 split-horizon
 !
End
```

Feature Information for MC-LAG TCN Interworking

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

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

Table 29: Feature Information for MC-LAG TCN Interworking

Feature Name	Releases	Feature Information
MC-LAG TCN Interworking	Cisco IOS XE Release 3.17S	Multiple VLAN Registration Protocol (MVRP) is used for MAC Flushing during the Pseudowire (PW) redundancy process. However, not all Dual Homed Device (DHD) switches support MVRP for MAC flushing. MC-LAG TCN Interworking feature enables using the Multiple Spanning Tree Protocol with Topology Change Notification (MSTP TCN) scheme for MAC Flushing towards the access network. The following commands were introduced or modified: **mlacp mac mstp-tcn**, **show ethernet service**, **show spanning-tree detail**

CHAPTER 20

Configuring ITU-T Y.1731 Fault Management Functions in IEEE CFM

This document describes the implementation of the ITU-Y.1731 fault management functions Ethernet Alarm Indication Signal (ETH-AIS) and Ethernet Remote Defect Indication (ETH-RDI) as part of the IEEE Ethernet Connectivity Fault Management (CFM) protocol.

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.

Prerequisites for Configuring ITU-T Y.1731 Fault Management Functions

Business Requirements

- Business and service policies have been established.
- Network topology and network administration have been evaluated.

Technical Requirements

- CFM must be configured and enabled for Y.1731 fault management features to function.
- A server maintenance endpoint (SMEP) is needed to support the ETH-AIS function.
- Maintenance intermediate points (MIPs) must be configured to support AIS messages; they are generated only on an interface on which a MIP is configured.

Restrictions for Configuring ITU-T Y.1731 Fault Management Functions

- Because of a port-ASIC hardware limitation, IEEE CFM cannot coexist with the Per VLAN Spanning Tree (PVST) protocol, and IEEE CFM cannot operate with the following line cards on the same system:
    - FI_WS_X6196_RJ21
    - FI_WS_X6196_RJ45
    - FI_WS_X6548_RJ21
    - FI_WS_X6548_RJ45
- CFM loopback messages are not confined within a maintenance domain according to their maintenance level. The impact of not having CFM loopback messages confined to their maintenance levels occurs at these levels:
    - Architecture--CFM layering is violated for loopback messages.
    - Deployment--A user may misconfigure a network and have loopback messages succeed.
    - Security--A malicious device that recognizes devices' MAC addresses and levels may explore a network topology that should be transparent.
- Routed interfaces are supported only in Cisco IOS Release 12.4(11)T.
- IEEE CFM is not fully supported on a Multiprotocol Label Switching (MPLS) provider edge (PE) device. There is no interaction between IEEE CFM and an Ethernet over MPLS (EoMPLS) pseudowire. A CFM packet can be transparently passed like regular data packets only via pseudowire, with the following restriction:
    - For policy feature card (PFC)-based EoMPLS, which uses a Cisco Catalyst LAN card as the MPLS uplink port, a CFM packet can be transparently passed via an EoMPLS pseudowire the same way regular data packets are passed. The EoMPLS endpoint interface, however, cannot be a maintenance endpoint (MEP) or an MIP, although a CFM MEP or MIP can be supported on regular Layer 2 switchport interfaces.
- CFM configuration is not supported on an EtherChannel in FastEthernet Channel (FEC) mode.

Information About Configuring ITU-T Y.1731 Fault Management Functions

Continuity Check Messages

CFM continuity check messages (CCMs) are multicast heartbeat messages exchanged periodically among MEPs. CCMs allow MEPs to discover other MEPs within a domain and allow MIPs to discover MEPs. CCMs are confined to a domain.

For more information about CCMs, see the “Continuity Check Messages” section of the "Configuring IEEE Standard-Compliant Ethernet CFM in a Service Provider Network" configuration module.

Server MEPs

Server MEPs (SMEPs) are virtual MEPs that perform two functions--server layer termination for CFM maintenance associations defined at a link or at the transport layer and server-Ethernet adaptation. When a SMEP detects a defect at the server layer, it issues frames containing ETH-AIS information.

Defect Conditions Detected by a MEP

The defect conditions that a MEP detects and subsequently acts upon are the following:

- AIS condition--A MEP receives an AIS frame.
- Dying gasp--An unrecoverable and vendor-specific condition. Dying gasp is generated in the following conditions:
  - Administratively disabling 802.3ah
  - Link down caused by administration down
  - Power failure
  - Reload

Note Administratively disabling 802.3ah does not disrupt traffic and should not generate an AIS. If a Reason field is empty, however, disabling always generates an AIS when Cisco routers and non-Cisco routers are interworking.

A notification about the defect condition may be sent immediately and continuously.

- Loss of continuity (LOC) condition--A MEP stops receiving CCMs from a peer MEP. An LOC condition is a MEP down error.

LOC results when a remote MEP lifetime timer expires and causes an AIS condition for the local MEP. The LOC condition is cleared when connectivity is restored.

- Mismerge condition--A CCM with a correct maintenance level but incorrect maintenance ID indicates that frames from a different service instance are merged with the service instance represented by the receiving MEP's maintenance ID. A mismerge condition is a cross-connect error.
- RDI condition--A MEP receives a CCM with the RDI field set.
- Signal fail condition--Declared by a MEP or the server layer termination function to notify the SMEP about a defect condition in the server layer. Signal fail conditions are as follows:
    - Configuration error
    - Cross-connect error
    - LOC
    - Loop error
    - MEP missing
    - MEP unknown (same as unexpected MEP)

Signal fail conditions cause AIS defect conditions for the MEP, resulting in the MEP receiving an AIS frame.

A MEP that detects a signal fail condition sends AIS frames to each of the client layer or sublayer maintenance associations.

- Unexpected MEP condition--A CCM with a correct maintenance level, correct maintenance ID, and an unexpected maintenance point ID (MPID) that is the same as the receiving MEP's MPID. An unexpected MEP condition is either a cross-check error or a configuration error.

Determination of an unexpected MPID is possible when a MEP maintains a list of its peer MPIDs. Peer MPIDs must be configured on each MEP during provisioning.

ETH-AIS Function

The ETH-AIS function suppresses alarms when a defect condition is detected at either the server layer or the server sublayer (virtual MEP). Transmission of frames carrying ETH-AIS information can be either enabled or disabled on either a MEP or a SMEP and can be sent at the client maintenance level by either a MEP or SMEP when a defect condition is detected.

SMEPs monitor the entire physical link so that an AIS is generated for each VLAN or server on the network. MEPs monitor VLANs, Ethernet virtual circuits (EVCs), and SMEPs where link up or link down and 802.3ah interworking are supported. A MEP that detects a connectivity fault at a specific level multicasts an AIS in the direction opposite the detected failure at the client maintenance association (MA) level.

An AIS causes a receiving MEP to suppress traps to prevent the network management system (NMS) from receiving an excessive number of redundant traps and also so that clients are asynchronously informed about faults.

In a point-to-point topology, a MEP has a single peer MEP and there is no ambiguity regarding the peer MEP for which it should suppress alarms when it receives ETH-AIS information.

In a multipoint Ethernet topology, a MEP that receives a frame with ETH-AIS information cannot determine which remote peer lost connectivity. The MEP also cannot determine the associated subset of peer MEPs for which it should suppress alarms because the ETH-AIS information does not include that MEP information. Because the MEP cannot determine the affected peer MEPs, it suppresses alarms for all peer MEPs whether or not there is connectivity.

Due to independent restoration capabilities within Spanning Tree Protocol (STP) environments, ETH-AIS is not expected to be applied in these environments; however, ETH-AIS transmission is configurable in STP environments by a network administrator.

ETH-AIS Transmission Reception and Processing

Only a MEP or a SMEP can be configured to send frames with ETH-AIS information. When a MEP detects a defect condition, it immediately begins transmitting frames with ETH-AIS information at the configured client maintenance level, which is the level at which the MIP is configured on the interface. Frames are transmitted to peer MEPs in the direction opposite the fault. The first AIS frame must always be transmitted immediately following the detection of a defect condition, but thereafter frames are transmitted at a frequency based on the configured AIS transmission period. The transmitting MEP continues to transmit frames with ETH-AIS information until the defect condition is removed. The period flag in the frame's header indicates the transmission interval. The default is that a MEP clears a defect condition only if no AIS frames are received within a time period equal to 3.5 times the configured transmission interval.

Note An AIS transmission period of one second is recommended; however, an AIS transmission period of one minute is supported to enable ETH-AIS across all VLANs supported by IEEE CFM.

When a MEP receives a frame with ETH-AIS information, it examines the frame to ensure that the maintenance association level corresponds to its own maintenance association level. The MEP detects the AIS condition and suppresses loss-of-continuity alarms associated with all its peer MEPs. Peer MEPs can resume generating loss-of-continuity alarms only when the receiving MEP exits the AIS condition.

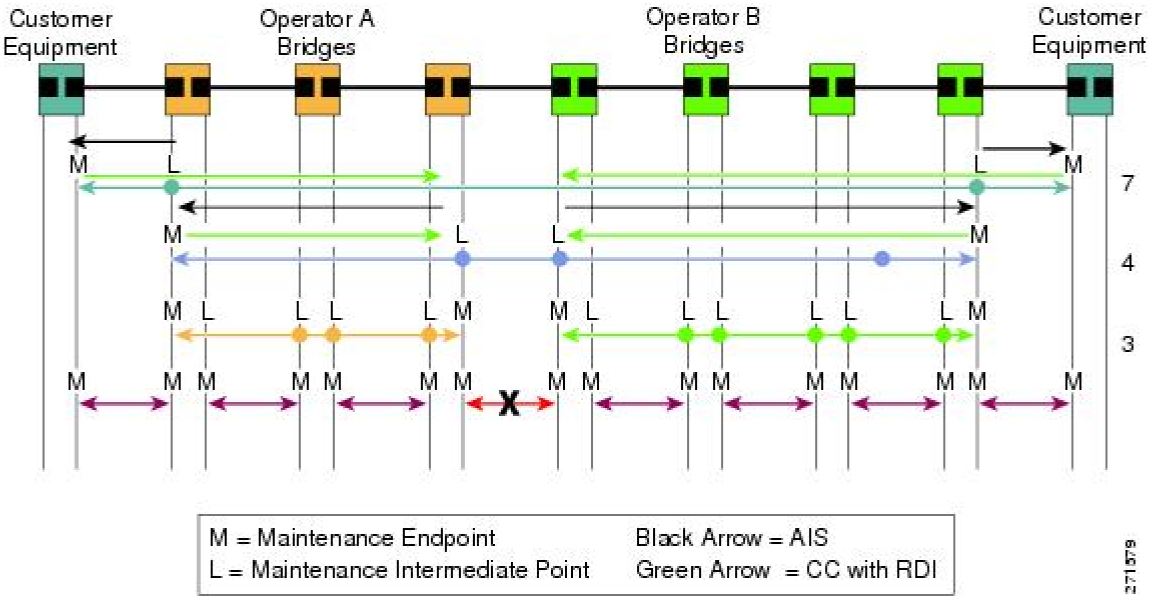
The client layer or client sublayer may consist of multiple maintenance associations that should also be notified to suppress alarms when either a server layer or server sublayer MEP detects a defect condition. The first AIS frame for all client layer or sublayer maintenance associations must be transmitted within one second after the defect condition is detected.

AIS and 802.3ah Interworking

The following conditions impact SMEP AIS conditions:

- By default, link down events cause the SMEP to enter the AIS condition and generate AIS frames for all services at the immediate client maintenance association level.
- Link up events cause the SMEP to exit the AIS state and stop generating AIS frames.
- Local fault detection results from dying gasp, link fault, or critical 802.3ah Remote Fault Indication (RFI). When 802.3ah is reestablished, the SMEP exits the AIS state and stops generating AIS frames.
- Local fault detection due to crossing of a high threshold with a configurable action of error disabling the interface.
- RFI received from a dying gasp, link fault, or critical event.

If a detected fault is due to dying gasp, the link goes down in both directions, creating AIS and RDI frame flow as shown in the figure below.



ETH-RDI Function

The ETH-RDI function is used by a MEP to communicate to its peer MEPs that a defect condition has been encountered. ETH-RDI is used only when ETH-CC transmission is enabled.

ETH-RDI has the following two applications:

- Single-ended fault management--A receiving MEP detects an RDI defect condition, which is correlated with other defect conditions in the MEP and may become the cause of a fault. If ETH-RDI information is not received by a single MEP, there are no defects in the entire MA.
- Contribution to far-end performance monitoring--A defect condition in the far end is used as an input to the performance monitoring process.

A MEP in a defect condition transmits CCMs with ETH-RDI information. A MEP that receives a CCM examines it to ensure that its maintenance association level corresponds to its configured maintenance association level and detects the RDI condition if the RDI field is set. The receiving MEP sets the RDI field in CCMs for the duration of a defect condition, and if the MEP is enabled for CCM transmission, transmits CCMs based on the configured transmission interval. When the defect condition clears, the MEP clears the RDI field in CCMs for subsequent transmissions.

In a point-to-point Ethernet connection, a MEP can clear an RDI condition when it receives the first CCM with the RDI field cleared from its peer MEP. In a multipoint Ethernet connection, a MEP cannot determine the peer MEP with the default condition and can clear an RDI condition only when it receives a CCM with the RDI field cleared from each of its peer MEPs.

The ETH-RDI function is part of continuity checking and is enabled by default. For more information about continuity checking, see the "Configuring IEEE Standard-Compliant Ethernet CFM in a Service Provider Network" configuration module.

How to Configure ITU-T Y.1731 Fault Management Functions

ETH-AIS and ETH-RDI both are enabled by default when CFM is configured, but each can also be manually enabled by a separate command during CFM configuration. Perform these tasks to either disable or enable the functions.

Disabling the ETH-AIS Function

Perform this task to disable the ETH-AIS function.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm ais link-status global**
4. **disable**
5. **exit**
6. **ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**]
7. **service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]]
8. **no ais** [**expiry-threshold** | **level** | **period** | **suppress-alarms**]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm ais link-status global** **Example:** `Device(config)# ethernet cfm ais link-status global`	Globally enables AIS generation and enters CFM SMEP AIS configuration mode.
Step 4	**disable** **Example:** `Device(config-ais-link-cfm)# disable`	Disables AIS transmission.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-ais-link-cfm)# exit`	Returns the CLI to global configuration mode.
Step 6	**ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**] **Example:** `Device(config)# ethernet cfm domain PROVIDERDOMAIN level 4`	Defines a CFM maintenance domain at a particular maintenance level and enters Ethernet CFM configuration mode.
Step 7	**service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]] **Example:** `Device(config-ecfm)# service customer101provider evc customer101provider@101 vlan 101`	Configures a maintenance association within a maintenance domain and enters Ethernet CFM service configuration mode.
Step 8	**no ais** [**expiry-threshold** | **level** | **period** | **suppress-alarms**] **Example:** `Device(config-ecfm-srv)# no ais`	Disables the AIS function for a specific maintenance association.

Step 9	**end**
	Example:
	`Device(config-ecfm-srv)# end`
		Returns the CLI to privileged EXEC mode.

Enabling ETH-AIS for a Single Interface SMEP and Disabling ETH-AIS for All Other Ports

Perform this task to manually enable the ETH-AIS function.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**]
4. **service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]]
5. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
6. **ais** [**expiry-threshold** *threshold* | **level** *level-id* | **period** *seconds*| **suppress-alarms**]
7. **ais** [**expiry-threshold** *threshold* | **level** *level-id* | **period** *seconds*| **suppress-alarms**]
8. **exit**
9. **service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]]

10. **continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**]
11. **ethernet cfm ais link-status global**
12. disable
13. **interface** *type number*
14. **ethernet oam remote-loopback** {**supported** | **timeout** *seconds*}
15. **ethernet cfm mip level** *level-id* [**vlan** {*vlan-id*| *vlan-id* - *vlan-id*| **,** *vlan-id* - *vlan-id*}]
16. **ethernet cfm ais link-status** [**level** *level-id*| **period** *seconds*]
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ethernet cfm domain** *domain-name* **level** *level-id* [**direction outward**] **Example:** `Device(config)# ethernet cfm domain PROVIDERDOMAIN level 4`	Defines a CFM maintenance domain at a particular maintenance level and enters Ethernet CFM configuration mode.
Step 4	**service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]] **Example:** `Device(config-ecfm)# service customer101provider evc customer101provider@101 vlan 101`	Configures a maintenance association within a maintenance domain and enters Ethernet CFM service configuration mode.
Step 5	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:** `Device(config-ecfm-srv)# continuity-check`	Enables the transmission of CCMs.
Step 6	**ais** [**expiry-threshold** *threshold* | **level** *level-id* | **period** *seconds*| **suppress-alarms**] **Example:** `Device(config-ecfm-srv)# ais period 1`	Enables the AIS function for a specific maintenance association.

	Command or Action	Purpose
Step 7	**ais** [**expiry-threshold** *threshold* | **level** *level-id* | **period** *seconds*| **suppress-alarms**] **Example:** `Device(config-ecfm-srv)# ais level 7`	Enables the AIS function for a specific maintenance association.
Step 8	**exit** **Example:** `Device(config-ecfm-srv)# exit`	Returns the CLI to Ethernet CFM configuration mode.
Step 9	**service** {*ma-name* | *ma-num* | **vlan-id** *vlan-id* | **vpn-id** *vpn-id*} [**port** | **vlan** *vlan-id* [**direction down**]] **Example:** `Device(config-ecfm)# service customer110provider evc customer110provider@110 vlan 110`	Configures a maintenance association within a maintenance domain and enters Ethernet CFM service configuration mode.
Step 10	**continuity-check** [**interval** *time* | **loss-threshold** *threshold* | **static rmep**] **Example:** `Device(config-ecfm-srv)# continuity-check`	Enables the transmission of CCMs.
Step 11	**ethernet cfm ais link-status global** **Example:** `Device(config-ecfm-srv)# ethernet cfm ais link-status global`	Globally enables AIS generation and places the CLI in CFM SMEP AIS configuration mode (config-ais-link-cfm) to configure AIS commands for a SMEP.
Step 12	disable **Example:** `Device(config-ais-link-cfm)# disable`	Disables the generation of AIS frames resulting from a link-status change.
Step 13	**interface** *type* *number* **Example:** `Device(config-ais-link-cfm)# interface ethernet 0/1`	Configures an interface type and enters interface configuration mode.

Step 14	**ethernet oam remote-loopback** {**supported** | **timeout** *seconds*}
	Example:
	`Device(config-if)# ethernet oam remote-loopback supported`
		Enables the support of Ethernet OAM remote loopback operations on an interface or sets a remote loopback timeout period.

	Command or Action	Purpose
Step 15	**ethernet cfm mip level** *level-id* [**vlan** {*vlan-id*| *vlan-id* - *vlan-id*| **,** *vlan-id* - *vlan-id*}] **Example:** `Device(config-if)# ethernet cfm mip level 4 vlan 101`	Provisions a MIP at a specified maintenance level on an interface.
Step 16	**ethernet cfm ais link-status** [**level** *level-id*| **period** *seconds*] **Example:** `Device(config-if)# ethernet cfm ais link-status`	Enables AIS generation from a SMEP.

Step 17	**end**
	Example:
	`Device(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Configuration Examples for Configuring ITU-T Y.1731 Fault Management Functions

Example: Enabling IEEE CFM on an Interface

The following example shows how to enable IEEE CFM on an interface:

```
!
ethernet cfm domain ServiceProvider level 4
mep archive-hold-time 60
service MetroCustomer1 vlan 100
!
ethernet cfm domain OperatorA level 1
mep archive-hold-time 65
service MetroCustomer1OpA vlan 100
!
ethernet cfm enable
ethernet cfm traceroute cache
ethernet cfm traceroute cache size 200
ethernet cfm traceroute cache hold-time 60
!
interface gigabitethernet3/0
ethernet cfm mip level 1
!
interface gigabitethernet4/0
ethernet cfm mip level 4
ethernet cfm mep level 1 mpid 102 vlan 100
!
ethernet cfm cc enable level 1 vlan 100
ethernet cfm cc level any vlan any interval 20 loss-threshold 3
```

Example: Enabling AIS

The following example shows how to enable AIS:

```
!
ethernet cfm domain PROVIDER_DOMAIN level 4
 service customer101provider evc customer101provider@101 vlan 101
  continuity-check
  ais period 1
  ais level 7
 service customer110provider evc customer110provider@110 vlan 110
  continuity-check
!
ethernet cfm ais link-status global
 disable
!
!
interface Ethernet 0/1
 no ip address
 ethernet oam remote-loopback supported
 ethernet oam
 ethernet cfm mip level 4 vlan 1,101,110
 ethernet cfm ais link-status
!
```

Example: Show Commands Output

The following sample output from the **show ethernet cfm maintenance-point local detail** command shows the settings for the local MEP:

```
Device# show ethernet cfm maintenance-points local detail

MEP Settings:
-------------
MPID: 2101
DomainName: PROVIDERDOMAIN
Level: 4
Direction: I
Vlan: 101
Interface: Et0/1
CC-Status: Enabled
MAC: aabb.cc03.8410
Defect Condition: AIS
presentRDI: TRUE
AIS-Status: Enabled
AIS Period: 1000(ms)
AIS Expiry Threshold: 3.5
Level to transmit AIS: Default
Suppress Alarm configuration: Enabled
Suppressing Alarms: Yes
```

The following sample output from the **show ethernet cfm smep** command shows the settings for a SMEP:

```
Device# show ethernet cfm smep
SMEP Settings:
--------------
Interface: Ethernet0/0
AIS-Status: Enabled
AIS Period: 60000 (ms)
```

```
Level to transmit AIS: 4
Defect Condition: No Defect
```

The following sample output from the **show ethernet cfm smep interface** command shows the settings for a specific interface on a SMEP:

```
Device# show ethernet cfm smep interface ethernet 0/1
SMEP Settings:
--------------
Interface: Ethernet0/1
LCK-Status: Enabled
LCK Period: 60000 (ms)
Level to transmit LCK: Default
AIS-Status: Enabled
AIS Period: 60000 (ms)
Level to transmit AIS: Default
Defect Condition: No Defect
Router#
```

The following sample output from the **show ethernet cfm errors** command shows the Ethernet CFM errors on a device:

```
Device# show ethernet cfm errors
Level    Vlan    MPID    Remote MAC        Reason         Service ID
5        102     -       aabb.cc00.ca10    Receive AIS    service test
```

The following sample output from the **show ethernet cfm maintenance-points remote detail** command shows the detailed information about a specific remote MEP:

```
Device# show ethernet cfm maintenance-points remote detail mpid 66
MAC Address: aabb.cc00.ca10
Domain/Level: PROVIDERDOMAIN/4
EVC: test
MPID: 66 (Can ping/traceroute)
Incoming Port(s): Ethernet0/2
CC Lifetime(sec): 75
Age of Last CC Message(sec): 8
Receive RDI: TRUE
Frame Loss: 0%
CC Packet Statistics: 2/0 (Received/Error)
R1#MAC Address: aabb.cc00.ca10
Domain/Level: PROVIDERDOMAIN/4
EVC: test
MPID: 66 (Can ping/traceroute)
Incoming Port(s): Ethernet0/2
CC Lifetime(sec): 75
Age of Last CC Message(sec): 8
Receive RDI: TRUE
Frame Loss: 0%
CC Packet Statistics: 2/0 (Received/Error)
```

Additional References

Related Documents

Related Topic	Document Title
IEEE CFM	"Configuring IEEE Standard-Compliant Ethernet CFM in a Service Provider Network"
Using OAM	"Using Ethernet Operations, Administration, and Maintenance"
IEEE CFM and Y.1731 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*

Standards

Standard	Title
IEEE 802.1ag	*802.1ag - Connectivity Fault Management*
IEEE 802.3ah	*Ethernet in the First Mile*
ITU-T	*ITU-T Y.1731 OAM Mechanisms for Ethernet-Based Networks*

Technical Assistance

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

Feature Information for Configuring ITU-T Y.1731 Fault Management Functions

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

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

Table 30: Feature Information for Configuring ITU-T Y.1731 Fault Management Functions

Feature Name	Releases	Feature Information
Configuring ITU-T Y.1731 Fault Management Functions	15.0(1)XA 12.2(33)SRE 15.1(1)T Cisco IOS XE Release 3.8S	The ITU-Y.1731 Fault Management Functions feature adds to IEEE CFM the ETH-AIS and ETH-RDI functions for fault detection, fault verification, and fault isolation in large MANs and WANs. The following commands were introduced or modified: **ais**, **clear ethernet cfm ais**, **disable**(CFM-AIS-link), **ethernet cfm ais link-status**, **ethernet cfm ais link-status global**, **level**(cfm-ais-link), **period**(cfm-ais-link), **show ethernet cfm errors**, **show ethernet cfm maintenance-points local**, **show ethernet cfm maintenance-points remote detail**, **show ethernet cfm smep**.

CHAPTER 21

Configuring IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations

This module describes how to configure an IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) operation to gather the following performance measurements for Ethernet service:

- Ethernet Delay
- Ethernet Delay Variation
- Ethernet Frame Loss Ratio

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.

Prerequisites for ITU-T Y.1731 Operations

IEEE-compliant Connectivity Fault Management (CFM) must be configured and enabled for Y.1731 performance monitoring to function.

Note Y1731 is supported on Port Channel interfaces.

Restrictions for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731)

- SNMP is not supported for reporting threshold events or collecting performance statistics for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) operations.

  SNMP is partially supported; the results for DM/LM can be polled for some attributes. However MIB support for all parameters is not supported.
- Continuity Check Message (CCM)-based dual-ended Ethernet frame loss operations are not supported.
- In a single-ended Ethernet operation, performance measurement statistics can be retrieved only at the device on which the sender Ethernet Connectivity Fault Management (CFM) Maintenance End Point (MEP) is configured.
- To avoid losing the CoS value configured on the frames, do not configure **rewrite** on the EFPs throughout the Layer2 circuit. The CoS value is preserved, if the Y.1731 frames are marked with specific CoS value.
- CFM over cross-connect on the routers works only if the **control-word** is configured. To start DM timestamping, switch ON the control-word if the remote end is not switched ON.
- To avoid errors in RX and TX timestamping, ensure to have Y1731 sender as primary PTP, and the Y1731 responder as subordinate PTP.
- Reconfigure IP SLA Y1731 while doing online insertion removal (OIR) of IM or router reload because local MEP is deleted during the course.
- A delay may be observed after issuing the **ip sla schedule** command after a reload of the router is performed, to populate with the Y.1731 PM measurements.
- The dot1q tag contains class of service (CoS) bits, which are used by IPSLA Y.1731 PM session to test delay or loss of packets with a specific CoS. This CoS cannot be a non-zero value when using EPM over untagged EFPs.

How to Configure IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations

Configuring a Dual-Ended Ethernet Delay or Delay Variation Operation

Perform the tasks for configuring a dual-ended operation in the order presented.

Note To remove the MEP configurations in an already-configured dual-ended operation, always remove the MEPs in the reverse order in which they were configured. That is, remove the scheduler first, then the threshold monitoring configuration, and then the sender MEP configuration on the source device before removing the scheduler, proactive threshold monitoring, and receiver MEP configuration on the destination device.

Configuring a Receiver MEP on the Destination Device

Before you begin

Time synchronization is required between the source and destination devices in order to provide accurate one-way delay (latency) or delay-variation measurements. Configure either Precision Time Protocol (PTP) or Network Time Protocol (NTP) on both the source and destination devices.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip sla** *operation-number*
4. **ethernet y1731 delay receive 1DM domain** *domain-name* {**evc** *evc-id* | **vlan** *vlan-id*} **cos** *cos* {**mpid** *source-mp-id* | **mac-address** *source-address*}
5. **aggregate interval** *seconds*
6. **distribution** {**delay** | **delay-variation**} **one-way** *number-of-bins boundary*[,...,*boundary*]
7. **frame offset** *offset-value*
8. **history interval** *intervals-stored*
9. **max-delay** *milliseconds*
10. **owner** *owner-id*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip sla** *operation-number* **Example:** `Router(config-term)# ip sla 501`	Begins configuring an IP SLAs operation and enters IP SLA configuration mode.

	Command or Action	Purpose

Step 4	**ethernet y1731 delay receive 1DM domain** *domain-name* {**evc** *evc-id* | **vlan** *vlan-id*} **cos** *cos* {**mpid** *source-mp-id* | **mac-address** *source-address*}
	Example:
	`Router(config-ip-sla)# ethernet y1731 delay receive 1DM domain xxx evc yyy cos 3 mpid 101`
		Begins configuring the receiver on the responder and enters IP SLA Y.1731 delay configuration mode.
		• The *source-mp-id* or *source-address* configured by this command corresponds to that of the MEP being configured.
		Note The session with mac-address will not be inactivated when there is CFM error.

Step 5	**aggregate interval** *seconds* **Example:** `Router(config-sla-y1731-delay)# aggregate interval 900`	(Optional) Configures the length of time during which the performance measurements are conducted and the results stored.
Step 6	**distribution** {**delay** | **delay-variation**} **one-way** *number-of-bins boundary*[**,...,***boundary*] **Example:** `Router(config-sla-y1731-delay)# distribution delay-variation one-way 5 5000,10000,15000,20000,-1`	(Optional) Specifies measurement type and configures bins for statistics distributions kept.
Step 7	**frame offset** *offset-value* **Example:** `Router(config-sla-y1731-delay)# frame offset 1`	(Optional) Sets the value for calculating delay variation rates.
Step 8	**history interval** *intervals-stored* **Example:** `Router(config-sla-y1731-delay)# history interval 2`	(Optional) Sets the number of statistics distributions kept during the lifetime of an IP SLAs Ethernet operation.
Step 9	**max-delay** *milliseconds* **Example:** `Router(config-sla-y1731-delay)# max-delay 5000`	(Optional) Sets the amount of time an MEP waits for a frame.
Step 10	**owner** *owner-id* **Example:** `Router(config-sla-y1731-delay)# owner admin`	(Optional) Configures the owner of an IP SLAs operation.

	Command or Action	Purpose

Step 11	**end**
	Example:
	`Router(config-sla-y1731-delay)# end`
		Exits to privileged EXEC mode.

What to do next

To add proactive threshold conditions and reactive triggering for generating traps, see the "Configuring Proactive Threshold Monitoring" module of the *IP SLAs Configuration Guide*.

When you are finished configuring proactive threshold monitoring for this MEP, see the "Scheduling IP SLAs Operations" section to schedule the operation.

Configuring the Sender MEP on the Source Router

Before you begin

- Time synchronization is required between the source and destination devices in order to provide accurate one-way delay (latency) or delay-variation measurements. Configure either Precision Time Protocol (PTP) or Network Time Protocol (NTP) on both the source and destination devices.
- The receiver MEP must be configured, including proacive threshold monitoring, and scheduled before you configure the sender MEP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip sla operation-number**
4. **ethernet y1731 delay 1DM 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*}}
5. **aggregate interval** *seconds*
6. **frame interval** *milliseconds*
7. **frame size** *bytes*
8. **history interval** *intervals-stored*
9. **owner** *owner-id*
10. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip sla operation-number** **Example:** `Router(config)# ip sla 500`	Begins configuring an IP SLAs operation and enters IP SLA configuration mode.

Step 4	**ethernet y1731 delay 1DM 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 delay 1DM domain xxx evc yyy mpid 101 cos 3 source mpid 100`
		Begins configuring a dual-ended Ethernet delay operation and enters IP SLA Y.1731 delay configuration mode.
		Note The session with mac-address will not be inactivated when there is CFM error.

Step 5	**aggregate interval** *seconds* **Example:** `Router(config-sla-y1731-delay)# aggregate interval 900`	(Optional) Configures the length of time during which the performance measurements are conducted and the results stored.
Step 6	**frame interval** *milliseconds* **Example:** `Router(config-sla-y1731-delay)# frame interval 100`	(Optional) Sets the gap between successive frames.
Step 7	**frame size** *bytes* **Example:** `Router(config-sla-y1731-delay)# frame size 64`	(Optional) Sets the padding size for frames.
Step 8	**history interval** *intervals-stored* **Example:** `Router(config-sla-y1731-delay)# history interval 2`	(Optional) Sets the number of statistics distributions kept during the lifetime of an IP SLAs Ethernet operation.

	Command or Action	Purpose
Step 9	**owner** *owner-id* **Example:** `Router(config-sla-y1731-delay)# owner admin`	(Optional) Configures the owner of an IP SLAs operation.

Step 10	**end**
	Example:
	`Router(config-sla-y1731-delay)# end`
		Exits to privileged EXEC mode.

What to do next

To add proactive threshold conditions and reactive triggering for generating traps, see the "Configuring Proactive Threshold Monitoring" module of the *IP SLAs Configuration Guide*.

When you are finished configuring proactive threshold monitoring for this MEP, see the "Scheduling IP SLAs Operations" section to schedule the operation.

Configuring a Sender MEP for a Single-Ended Ethernet Delay or Delay Variation Operation

Perform this task to configure a sender MEP on the source device.

Before you begin

- Time synchronization is required between the source and destination devices in order to provide accurate one-way delay (latency) or delay-variation measurements. Configure either Precision Time Protocol (PTP) or Network Time Protocol (NTP) on both the source and destination devices.

Note To display information about remote (target) MEPs on destination devices, use the **show ethernet cfm maintenance-points remote** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip sla** *operation-number*
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*}}
5. **clock sync**
6. **aggregate interval** *seconds*
7. **distribution** {**delay** | **delay-variation**} **one-way** *number-of-bins boundary*[**,...,***boundary*]

8. **frame interval** *milliseconds*
9. **frame offset** *offset-value*
10. **frame size** *bytes*
11. **history interval** *intervals-stored*
12. **max-delay** *milliseconds*
13. **owner** *owner-id*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip sla** *operation-number* **Example:** `Device(config-term)# ip sla 10`	Begins configuring an IP SLAs operation and enters IP SLA configuration mode.

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`
		Begins configuring a single-ended Ethernet delay operation and enters IP SLA Y.1731 delay configuration mode.
		• To configure concurrent operations, use the **DMMv1** keyword with this command. Repeat the preceding two steps to each concurrent operation, to be added to a single IP SLA operation number. Concurrent operations are supported for a given EVC, CoS, and remote MEP combination, or for multiple MEPs for a given multipoint EVC.
		Note The session with mac-address will not be inactivated when there is CFM error.

Step 5	**clock sync** **Example:** `Device(config-sla-y1731-delay)# clock sync`	(Optional) Indicates that the end points are synchronized and thus allows the operation to calculate one-way delay measurements.

	Command or Action	Purpose
Step 6	**aggregate interval** *seconds* **Example:** `Device(config-sla-y1731-delay)# aggregate interval 900`	(Optional) Configures the length of time during which the performance measurements are conducted and the results stored.
Step 7	**distribution** {**delay** | **delay-variation**} **one-way** *number-of-bins boundary*[**,...,***boundary*] **Example:** `Device(config-sla-y1731-delay)# distribution delay-variation one-way 5 5000, 10000,15000,20000,-1`	(Optional) Specifies measurement type and configures bins for statistics distributions kept.
Step 8	**frame interval** *milliseconds* **Example:** `Device(config-sla-y1731-delay)# frame interval 100`	(Optional) Sets the gap between successive frames.
Step 9	**frame offset** *offset-value* **Example:** `Device(config-sla-y1731-delay)# frame offset 1`	(Optional) Sets value for calculating delay variation values.
Step 10	**frame size** *bytes* **Example:** `Device(config-sla-y1731-delay)# frame size 32`	(Optional) Configures padding size for frames.
Step 11	**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.
Step 12	**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 13	**owner** *owner-id* **Example:**	(Optional) Configures the owner of an IP SLAs operation.

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

Step 14	**end**
	Example:
	`Device(config-sla-y1731-delay)# end`
		Exits to privileged EXEC mode.

What to do next

To add proactive threshold conditions and reactive triggering for generating traps, see the "Configuring Proactive Threshold Monitoring" module of the *IP SLAs Configuration Guide*.

When you are finished configuring proactive threshold monitoring for this operation, see the "Scheduling IP SLAs Operations" section to schedule the operation.

Configuring a Sender MEP for a Single-Ended Ethernet Frame Loss Ratio Operation

Note To display information about remote (target) MEPs on destination devices, use the **show ethernet cfm maintenance-points remote** command.

Perform this task to configure a sender MEP on the source device.

Before you begin

- Class of Service (CoS)-level monitoring must be enabled on MEPs associated to the Ethernet frame loss operation by using the **monitor loss counter** command on the devices at both ends of the operation. See the *Cisco IOS Carrier Ethernet Command Reference* for command information. See the "Configuration Examples for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations" section for configuration information.

  Note Cisco IOS Y.1731 implementation allows monitoring of frame loss for frames on an EVC regardless of the CoS value (any CoS or Aggregate CoS cases). See the "Configuration Examples for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations" section for configuration information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip sla** *operation-number*

4. **ethernet y1731 loss** {**LMM** | **SLM**} [**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*}}
5. **aggregate interval** *seconds*
6. **availability algorithm** {**sliding-window** | **static-window**}
7. **frame consecutive** *value*
8. **frame interval** *milliseconds*
9. **history interval** *intervals-stored*
10. **owner** *owner-id*
11. **exit**
12. **exit**
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip sla** *operation-number* **Example:** `Device(config-term)# ip sla 11`	Begins configuring an IP SLAs operation and enters IP SLA configuration mode.

Step 4	**ethernet y1731 loss** {**LMM** | **SLM**} [**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 loss LMM domain xxx vlan 12 mpid 34 CoS 4 source mpid 23`
		Begins configuring a single-ended Ethernet frame loss ratio operation and enters IP SLA Y.1731 loss configuration mode.
		• To configure concurrent operations, use the **SLM** keyword with this command. Repeat the preceding two steps to configure each concurrent operation to be added to a single IP SLA operation number. Concurrent operations are supported for a given EVC, CoS, and remote-MEP combination, or for multiple MEPs for a given multipoint EVC.
		Note The session with mac-address will not be inactivated when there is CFM error.

	Command or Action	Purpose
Step 5	**aggregate interval** *seconds* **Example:** `Device(config-sla-y1731-loss)# aggregate interval 900`	(Optional) Configures the length of time during which performance measurements are conducted and the results stored.

Step 6	**availability algorithm** {**sliding-window** | **static-window**}
	Example:
	`Device(config-sla-y1731-loss)# availability algorithm static-window`
		(Optional) Specifies availability algorithm used.

Step 7	**frame consecutive** *value* **Example:** `Device(config-sla-y1731-loss)# frame consecutive 10`	(Optional) Specifies number of consecutive measurements to be used to determine availability or unavailability status.
Step 8	**frame interval** *milliseconds* **Example:** `Device(config-sla-y1731-loss)# frame interval 100`	(Optional) Sets the gap between successive frames.
Step 9	**history interval** *intervals-stored* **Example:** `Device(config-sla-y1731-loss)# history interval 2`	(Optional) Sets the number of statistics distributions kept during the lifetime of an IP SLAs Ethernet operation.
Step 10	**owner** *owner-id* **Example:** `Device(config-sla-y1731-delay)# owner admin`	(Optional) Configures the owner of an IP SLAs operation.
Step 11	**exit** **Example:** `Device(config-sla-y1731-delay)# exit`	Exits to IP SLA configuration mode.
Step 12	**exit** **Example:**	Exits to global configuration mode.

	Command or Action	Purpose
	`Device(config-ip-sla)# exit`	
Step 13	**exit** **Example:** `Device(config)# exit`	Exits to privileged EXEC mode.

What to do next

When you are finished configuring this MEP, see the "Scheduling IP SLAs Operations" section to schedule the operation.

Scheduling IP SLAs Operations

Before you begin

- All IP Service Level Agreements (SLAs) operations to be scheduled must be already configured.
- The frequency of all operations scheduled in a multioperation group must be the same.
- The list of one or more operation ID numbers to be added to a multioperation group must be limited to a maximum of 125 characters in length, including commas (,).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Enter one of the following commands:
   - **ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [**start-time** {[*hh***:***mm***:***ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh***:***mm***:***ss*}] [**ageout** *seconds*] [**recurring**]
   - **ip sla group schedule** *group-operation-number operation-id-numbers* {**schedule-period** *schedule-period-range* | **schedule-together**} [**ageout** *seconds*] **frequency** *group-operation-frequency* [**life** {**forever** | *seconds*}] [**start-time** {*hh***:***mm* [**:***ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh***:***mm* [**:***ss*]}]
4. **end**
5. **show ip sla group schedule**
6. **show ip sla configuration**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Enter one of the following commands:
	• **ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [**start-time** {[*hh*:*mm*:*ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh*:*mm*:*ss*}] [**ageout** *seconds*] [**recurring**]
	• **ip sla group schedule** *group-operation-number operation-id-numbers* {**schedule-period** *schedule-period-range* | **schedule-together**} [**ageout** *seconds*] **frequency** *group-operation-frequency* [**life** {**forever** | *seconds*}] [**start-time** {*hh*:*mm* [:*ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh*:*mm* [:*ss*]}]
	Example:
	`Device(config)# ip sla schedule 10 life forever start-time now`
	`Device(config)# ip sla group schedule 10 schedule-period frequency`
	`Device(config)# ip sla group schedule 1 3,4,6-9 life forever start-time now`
	`Device(config)# ip sla schedule 1 3,4,6-9 schedule-period 50 frequency range 80-100`
		• Configures the scheduling parameters for an individual IP SLAs operation.
		• Specifies an IP SLAs operation group number and the range of operation numbers for a multioperation scheduler.

Step 4	**end**
	Example:
	`Device(config)# end`
		Exits global configuration mode and returns to privileged EXEC mode.

Step 5	**show ip sla group schedule** **Example:** `Device# show ip sla group schedule`	(Optional) Displays IP SLAs group schedule details.
Step 6	**show ip sla configuration** **Example:** `Device# show ip sla configuration`	(Optional) Displays IP SLAs configuration details.

Configuration Examples for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations

Example: Dual-Ended Ethernet Delay Operation

The following sample output shows the configuration, including default values, of a receiver MEP on the responder device for a dual-ended Ethernet delay or delay variation operation:

```
Device# show ip sla configuration 501

IP SLAs Infrastructure Engine-III
Entry number: 501
Owner: admin
Tag:
Operation timeout (milliseconds): 5000
Ethernet Y1731 Delay Operation
Frame Type: 1DM
Domain: xxx
ReceiveOnly: TRUE
Evc: yyy
Local Mpid: 101
CoS: 3
   Max Delay: 5000
Threshold (milliseconds): 5000
.
.
.
Statistics Parameters
  Aggregation Period: 900
  Frame offset: 1
  Distribution Delay One-Way:
   Number of Bins 10
   Bin Boundaries: 5000,10000,15000,20000,25000,30000,35000,40000,45000,-1
  Distribution Delay-Variation One-Way:
   Number of Bins 10
   Bin Boundaries: 5000,10000,15000,20000,25000,30000,35000,40000,45000,-1
History
  Number of intervals: 2
```

The following sample output shows the configuration, including default values, of the sender MEP for a dual-ended IP SLAs Ethernet delay or delay variation operation:

```
Device# show ip sla configuration 500

IP SLAs Infrastructure Engine-III
Entry number: 500
Owner:
Tag:
Operation timeout (milliseconds): 5000
Ethernet Y1731 Delay Operation
Frame Type: 1DM
Domain: yyy
ReceiveOnly: FALSE
Evc: xxx
Target Mpid: 101
Source Mpid: 100
```

```
CoS: 3
   Request size (Padding portion): 64
   Frame Interval: 1000
Threshold (milliseconds): 5000
.
.
.
Statistics Parameters
  Aggregation Period: 900
  Frame offset: 1
History
  Number of intervals: 22
```

Example: Frame Delay and Frame Delay Variation Measurement Configuration

The following sample output shows the performance monitoring session summary:

```
Device# show ethernet cfm pm session summary

Number of Configured Session : 2
Number of Active Session: 2
Number of Inactive Session: 0
```

The following sample output shows the active performance monitoring session:

```
Device# show ethernet cfm pm session active

Display of Active Session
--------------------------------------------------------------------------------
EPM-ID   SLA-ID    Lvl/Type/ID/Cos/Dir     Src-Mac-address Dst-Mac-address
--------------------------------------------------------------------------------
 0     10           3/BD-V/10/2/Down     d0c2.8216.c9d7  d0c2.8216.27a3
 1     11           3/BD-V/10/3/Down     d0c2.8216.c9d7  d0c2.8216.27a3
Total number of Active Session: 2

Device# show ethernet cfm pm session db 0

----------------------------------------------------------------------------
      TX Time FWD                   RX Time FWD
      TX Time BWD                   RX Time BWD              Frame Delay
      Sec:nSec                      Sec:nSec                  Sec:nSec
----------------------------------------------------------------------------
Session ID: 0
****************************************************************************
       234:526163572                  245:305791416
       245:306761904                  234:527134653            0:593
****************************************************************************
       235:528900628                  246:308528744
       246:309452848                  235:529825333            0:601
****************************************************************************
       236:528882716                  247:308511128
       247:309450224                  236:529822413            0:601
****************************************************************************
       237:526578788                  248:306207432
       248:307157936                  237:527529885            0:593
****************************************************************************
       238:527052156                  249:306681064
       249:307588016                  238:527959717            0:609
****************************************************************************
       239:526625044                  250:306254200
       250:307091888                  239:527463325            0:593
****************************************************************************
```

```
        240:528243204               251:307872648
        251:308856880               240:529228021           0:585
```

Example: Sender MEP for a Single-Ended Ethernet Delay Operation

The following sample output shows the configuration, including default values, of the sender MEP for a single-ended IP SLAs Ethernet delay operation:

```
Router# show ip sla configuration 10

IP SLAs Infrastructure Engine-III
Entry number: 10
Owner:
Tag:
Operation timeout (milliseconds): 5000
Ethernet Y1731 Delay Operation
Frame Type: DMM
Domain: xxx
Vlan: yyy
Target Mpid: 101
Source Mpid: 100
CoS: 4
   Max Delay: 5000
   Request size (Padding portion): 64
   Frame Interval: 1000
   Clock: Not In Sync
Threshold (milliseconds): 5000
.
.
.
Statistics Parameters
  Aggregation Period: 900
  Frame offset: 1
  Distribution Delay Two-Way:
   Number of Bins 10
   Bin Boundaries: 5000,10000,15000,20000,25000,30000,35000,40000,45000,-1
  Distribution Delay-Variation Two-Way:
   Number of Bins 10
   Bin Boundaries: 5000,10000,15000,20000,25000,30000,35000,40000,45000,-1
History
  Number of intervals: 2
```

Example: Sender MEP for a Single-Ended Ethernet Frame Loss Operation

The following output shows the configuration, including default values, of the sender MEP in a basic single-ended IP SLAs Ethernet frame loss ratio operation with a start-time of now:

```
Router# show ip sla configuration 11

IP SLAs Infrastructure Engine-III
Entry number: 11
Owner:
Tag:
Operation timeout (milliseconds): 5000
Ethernet Y1731 Loss Operation
Frame Type: LMM
Domain: xxx
```

```
Vlan: 12
Target Mpid: 34
Source Mpid: 23
CoS: 4
   Request size (Padding portion): 0
   Frame Interval: 1000
Schedule:
   Operation frequency (seconds): 60  (not considered if randomly scheduled)
   Next Scheduled Start Time: Start Time already passed
   Group Scheduled : FALSE
   Randomly Scheduled : FALSE
   Life (seconds): 3600
   Entry Ageout (seconds): never
   Recurring (Starting Everyday): FALSE
   Status of entry (SNMP RowStatus): ActiveThreshold (milliseconds): 5000
Statistics Parameters
  Aggregation Period: 900
  Frame consecutive: 10
  Availability algorithm: static-window
History
  Number of intervals: 2
```

Additional References for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations

Related Documents

Related Topic	Document Title
Cisco IOS Carrier Ethernet commands	Cisco IOS Carrier Ethernet Command Reference
Cisco IOS IP SLAs commands	Cisco IOS IP SLAs Command Reference
Ethernet CFM	"Configuring Ethernet Connectivity Fault Management in a Service Provider Network" module of the *Cisco IOS Carrier Ethernet Configuration Guide*
Network Time Protocol (NTP)	"Configuring NTP" module of the *Cisco IOS Network Management Configuration Guide*
Proactive threshold monitoring for Cisco IOS IP SLAs	"Configuring Proactive Threshold Monitoring of IP SLAs Operations" module of the *Cisco IOS IP SLAs Configuration Guide*

Standards and RFCs

Standard/RFC	Title
ITU-T Y.1731	*OAM functions and mechanisms for Ethernet-based networks*
No specific RFCs are supported by the features in this document.	--

MIBs

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

Technical Assistance

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

Feature Information for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) Operations

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

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

Table 31: Feature Information for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731)

Feature Name	Releases	Feature Information
IP SLA Support for ETH-SLM (Ethernet Synthetic Loss Measurement in Y1731)		Y.1731 Performance Monitoring (PM) provides a standard Ethernet PM function that includes measurement of Ethernet frame delay, frame delay variation, frame loss, and frame throughput measurements specified by the ITU-T Y-1731 standard and interpreted by the Metro Ethernet Forum (MEF) standards group.
Y1731 MIB Support through existing IPSLA MIBs		Support was added for reporting threshold events and collecting performance statistics for IP SLAs Metro-Ethernet 3.0 (ITU-T Y.1731) operations using SNMP.

CHAPTER 22

VXLAN-MCLAG Active-Active High Availability Support

The VXLAN-MCLAG Active-Active High Availability Support feature implements dual-home device with pseudo Multichassis Link Aggregation Control Protocol (pMLACP) redundancy mode and layer 2 VxLAN on the Cisco ASR1000 Series Aggregation Services Routers.

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 VXLAN-MCLAG Active-Active High Availability Support

- The loopback interface configured for this feature cannot be used for another feature.
- The loopback interface of NVE interface must be shut down before configuring pmLACP, VxLAN and routing protocol.
- Bridge domain supports one VXLAN Network Identifier (VNI) Ethernet flow point (EFP) member only.
- Shutting the bridge domain affects status of the NVE interface, not the pseudo mLACP status.

Information About VXLAN-MCLAG Active-Active High Availability Support

Virtual Extensible LAN

Virtual Extensible LAN (VXLAN) is a network virtualization overlay technology that provides Layer 2 connectivity for workloads residing at noncontiguous points in the data center network. VXLAN enables flexibility by allowing workloads to be placed anywhere, along with the traffic separation required in a multitenant environment. VXLAN is an industry-standard protocol and uses underlay IP networks. It extends Layer 2 segments over a Layer 3 infrastructure to build Layer 2 overlay logical networks. It encapsulates Ethernet frames into IP User Data Protocol (UDP) headers and transports the encapsulated packets through the underlay network to the remote VXLAN tunnel endpoints (VTEPs) using the normal IP routing and forwarding mechanism.

Multichassis Link Aggregation Group

Multichassis Link Aggregation Group (MC-LAG) and Inter-chassis Communication Protocol (ICCP) enable a switch/router to use standard Ethernet Link Aggregation for device dual-homing, with active/standby redundancy. MC-LAG provides a mean to dual home a device (the dual homed device (DHD)) to two different peer devices (the Point of Attachment), allowing to have the benefits of node redundancy. Point of Attachment (PoA) nodes run Inter-chassis Communication Protocol (ICCP) to synchronize state & form a Redundancy Group (RG).

In VXLAN - MCLAG Active-Active High Availability support, both the PoA ports are placed in active/active mode with manual VLAN load balancing. It provides higher bandwidth utilization than Multichassis Link Aggregation Control Protocol (mLACP). It also allows maximum flexibility for the Provider Edge-Customer Edge (PE-CE) inter-operability for dual-homing redundancy and failover recovery. Active and standby PoA nodes are configured on the identical interfaces, that is, the same loopback IP address and interface as VTEP source interface, VLAN and VNI mapping, and so on.

How to Configure VXLAN-MCLAG Active-Active High Availability Support

Configuring Interchassis Redundancy Groups on PoA

To configure interchassis redundancy groups on PoA, perform the steps below.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **interchassis group** *group-id*

5. **member ip** *peer ip address*
6. **monitor peer [bfd | track]**
7. **mlacp node-id** *node id*
8. **backbone interface** *backbone if*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Configures the redundancy group.
Step 4	**interchassis group** *group-id* **Example:** `Device(config-red)# interchassis group 2`	Configures interchassis group.
Step 5	**member ip** *peer ip address* **Example:** `Device(config-r-ic)# member ip 172.168.40.24`	Specifies IP address to be assigned to a remote peer dialing in to the interface.
Step 6	**monitor peer [bfd | track]** **Example:** `Device(config-r-ic)# monitor peer bfd`	Specifies the the peer monitoring method.
Step 7	**mlacp node-id** *node id* **Example:** `Device(config-r-ic)# mlacp node-id 2`	Configures mLACP node ID.
Step 8	**backbone interface** *backbone if* **Example:** `Device(config-r-ic)# backbone interface Gi0/0/2`	Configures a backbone interface for the redundancy group.

Step 9	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Channel on PoA

To configure port channel on PoA, perform the steps below.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface Port-channel** *port channel number*
4. **negotiation**
5. **lacp fast-switchover**
6. **mlacp interchassis group** *rg id*
7. **mlacp mode active-active**
8. **mlacp load-balance primary vlan** *vlan-id*
9. **mlacp load-balance secondary vlan** *vlan-id*
10. **service instance** *id* **ethernet**
11. **encapsulation dot1q**
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface Port-channel** *port channel number* **Example:** `Device(config-if)# interface Port-channel 2`	Configures the interface for port channel.
Step 4	**negotiation** **Example:** `Device(config-if)# negotiation`	Configures auto negotiation mode.
Step 5	**lacp fast-switchover** **Example:** `Device(config-if)# lacp fast-switchover`	Specifies LACP Port Channel interface.
Step 6	**mlacp interchassis group** *rg id* **Example:** `Device(config-if)# mlacp interchassis group 2`	Configures mLACP peer PoA RG ID.

	Command or Action	Purpose
Step 7	**mlacp mode active-active** **Example:** `Device(config-if)# mlacp mode active-active`	Enables mLACP active-active POA redundancy.
Step 8	**mlacp load-balance primary vlan** *vlan-id* **Example:** `Device(config-if)# mlacp load-balance primary vlan 40`	Configures the list of primary VLANs that will be active and inactive on the given PoA.
Step 9	**mlacp load-balance secondary vlan** *vlan-id* **Example:** `Device(config-if)# mlacp load-balance secondary vlan 20`	Configures the list of secondary VLANs that will be active and inactive on the given PoA.
Step 10	**service instance** *id* **ethernet** **Example:** `Device(config-if-srv)# service instance 20 ethernet`	Configures service instance identifier.
Step 11	**encapsulation dot1q** **Example:** `Device(config-if-srv)# encapsulation dot1q 20`	Configures ethernet frame match criteria.

Step 12	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Vxlan Unicast Core Configuration on POA

To configure Vxlan Unicast Core Configuration on POA, perform the steps below.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bridge-domain** *id*
4. **member vni** *number*
5. **member Port-channel** *number* **service-instance** *id*
6. **exit**
7. **interface Loopback** *number*
8. **ip address**
9. **exit**
10. **interface nve**
11. **member vni** *number*
12. **ingress-replication** *IPV4 address*

13. **exit**
14. **source-interface Loopback** *id*
15. **no shutdown**
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**bridge-domain** *id* **Example:** `Device(config)# bridge-domain 20`	Configures the bridge domain ID.
Step 4	**member vni** *number* **Example:** `Device(config-bdomain)# member vni 7777`	Configures member virtual network identifier (VNI).
Step 5	**member Port-channel** *number* **service-instance** *id* **Example:** `Device(config-bdomain)# member Port-channel1 service-instance 20`	Configures port channel and service instance.
Step 6	**exit** **Example:** `Device(config-bdomain)# exit`	Exits bridge domain mode and returns to global configuration mode.
Step 7	**interface Loopback** *number* **Example:** `Device(config-if)# interface Loopback10`	Specifies a loopback interface.
Step 8	**ip address** **Example:** `Device(config-if)# ip address 77.1.1.1 255.255.255.255`	Configures IP address.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 10	**interface nve** **Example:** `Device(config)# interface nve1`	Configures network virtualization endpoint interface.
Step 11	**member vni** *number* **Example:** `Device(config-if)# member vni 7777`	Configures VNI information.
Step 12	**ingress-replication** *IPV4 address* **Example:** `Device(config-if-nve-vni)# ingress-replication 99.1.1.1`	Configures remote Peer IPV4 Address.
Step 13	**exit** **Example:** `Device(config-if-nve-vni)# exit`	Exits network virtualization endpoint interface configuration mode and returns to global configuration mode.
Step 14	**source-interface Loopback** *id* **Example:** `Device(config-if)# source-interface Loopback10`	Configures interface loopback.
Step 15	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface.

Step 16	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Vxlan Multicast Core Configuration on POA

To configure Vxlan Multicast Core Configuration on POA, perform the steps below.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bridge-domain** *id*
4. **member vni** *number*
5. **member Port-channel** *number* **service-instance** *id*
6. **exit**
7. **interface Loopback** *number*
8. **ip address**
9. **ip pim sparse-dense-mode**

10. **exit**
11. **interface nve**
12. **member vni** *number* **mcast-group** *address*
13. **source-interface Loopback**
14. **no shutdown**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**bridge-domain** *id* **Example:** `Device(config)# bridge-domain 20`	Configures the bridge domain ID.
Step 4	**member vni** *number* **Example:** `Device(config-bdomain)# member vni 7777`	Configures member virtual network identifier (VNI).
Step 5	**member Port-channel** *number* **service-instance** *id* **Example:** `Device(config-bdomain)# member Port-channel1 service-instance 20`	Configures port channel and service instance.
Step 6	**exit** **Example:** `Device(config-bdomain)# exit`	Exits bridge domain mode and returns to global configuration mode.
Step 7	**interface Loopback** *number* **Example:** `Device(config-if)# interface Loopback10`	Specifies a loopback interface.
Step 8	**ip address** **Example:** `Device(config-if)# ip address 77.1.1.1 255.255.255.255`	Configures IP address.
Step 9	**ip pim sparse-dense-mode** **Example:**	Enables PIM to operate in sparse or dense mode.

	Command or Action	Purpose
	`Device(config-if)# ip pim sparse-dense-mode`	
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface nve** **Example:** `Device(config)# interface nve1`	Configures network virtualization endpoint interface.
Step 12	**member vni** *number* **mcast-group** *address* **Example:** `Device(config-if)# member vni 7777 mcast-group 232.1.1.1`	Configures VNI information.
Step 13	**source-interface Loopback** **Example:** `Device(config-if)# source-interface Loopback10`	Configures interface loopback.
Step 14	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface.

Step 15	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Dual-homed Device

To configure dual-homed device, perform the steps below:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface Port-channel** *number*
4. **switchport** *group-id*
5. **switchport trunk encapsulation dot1q**
6. **switchport trunk allowed vlan 20-50**
7. **switchport mode trunk**
8. **exit**
9. **interface GigabitEthernet3/1**
10. **switchport**
11. **switchport trunk encapsulation dot1q**
12. **switchport trunk allowed vlan 20-50**

13. **switchport mode trunk**
14. **channel-group** *number mode*
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface Port-channel** *number* **Example:** `Device(config)# interface Port-channel1`	Configures ethernet channel of interfaces.
Step 4	**switchport** *group-id* **Example:** `Device(config-if)# switchport`	Sets the interface as an Ethernet interface.
Step 5	**switchport trunk encapsulation dot1q** **Example:** `Device(config-r-ic)# switchport trunk encapsulation dot1q`	Defines the encapsulation format as IEEE 802.1Q (dot1q) for the specified interface.
Step 6	**switchport trunk allowed vlan 20-50** **Example:** `Device(config-r-ic)# switchport trunk allowed vlan 20-50`	Specifies that only certain VLANs are allowed on the specified trunk.
Step 7	**switchport mode trunk** **Example:** `Device(config-r-ic)# switchport mode trunk`	Sets the interface as an Ethernet trunk port.
Step 8	**exit** **Example:** `Device(config-r-ic)# exit`	Exits interface mode and returns to global configuration mode
Step 9	**interface GigabitEthernet3/1** **Example:** `Device(config-if)# interface GigabitEthernet3/1`	Enters the interface configuration mode on the Gigabit Ethernet interface.

	Command or Action	Purpose
Step 10	**switchport** **Example:** `Device(config-if)# switchport`	Configures the interface port.
Step 11	**switchport trunk encapsulation dot1q** **Example:** `Device(config-if)# switchport trunk encapsulation dot1q`	Defines the encapsulation format as IEEE 802.1Q (dot1q) for the specified interface.
Step 12	**switchport trunk allowed vlan 20-50** **Example:** `Device(config-if)# switchport trunk allowed vlan 20-50`	Specifies that only certain VLANs are allowed on the specified trunk.
Step 13	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the interface as an Ethernet trunk port.
Step 14	**channel-group** *number mode* **Example:** `Device(config-if)# channel-group 1 mode active`	Configures the port in a channel group and sets the mode.

Step 15	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Verifying VXLAN-MCLAG Active-Active High Availability Support

To verify, perform the steps below.

SUMMARY STEPS

1. **show lacp internal**
2. **show nve interface nve1**
3. **show nve peers**
4. **show platform software ethernet fp ac bridge-domain binding**
5. **show bridge-domain 20**
6. **show lacp multi-chassis load-balance port-channel**
7. **show nve vni 11111 detail**
8. **show lacp multi load group**

DETAILED STEPS

Step 1 **show lacp internal**

Example:

```
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode

Channel group 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi0/0/0   SA      bndl      32768         0x1       0x1     0x1         0x3D

Channel group 2
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi0/0/1   SA      susp      32768         0x2       0x2     0x2         0x7D
```

Step 2 **show nve interface nve1**

Example:

```
Interface: nve1, State: Admin Up, Oper Up Encapsulation: Vxlan
source-interface: Loopback10 (primary:77.1.1.1 vrf:0)
```

Step 3 **show nve peers**

Example:

```
Interface  Peer-IP          VNI        Peer state
   nve1    99.1.1.1         7777
```

Step 4 **show platform software ethernet fp ac bridge-domain binding**

Example:

```
Forwarding Manager Bridge Domain Bindings

BD    Interface                      EFP DPIDB  SHG   STP  AOM id
-----------------------------------------------------------------------------
20    Port-channel1.EFP20            16908305   None FRWD 182, (created)
20    nve1.VNI7777                   16908307   None FRWD 268, (created)
40    Port-channel1.EFP40            16908306   None BLCK 258, (created)
40    nve2.VNI8888                   16908308   None FRWD 285, (created)
```

Step 5 **show bridge-domain 20**

Example:

```
FBridge-domain 20 (2 ports in all)
State: UP                    Mac learning: Enabled
Aging-Timer: 300 second(s)
    Port-channel1 service instance 20
    vni 7777
   AED MAC address    Policy  Tag       Age  Pseudoport
   0   0000.6177.0003 forward dynamic   300  nve1.VNI7777, VxLAN
                                             src: 77.1.1.1 dst: 99.1.1.1
   0   0000.6177.0009 forward dynamic   300  nve1.VNI7777, VxLAN
                                             src: 77.1.1.1 dst: 99.1.1.1
   0   0000.6177.0000 forward dynamic   300  nve1.VNI7777, VxLAN
                                             src: 77.1.1.1 dst: 99.1.1.1
   0   0000.1577.0009 forward dynamic   300  Port-channel1.EFP20
```

Step 6 **show lacp multi-chassis load-balance port-channel**

Example:

```
Interface Port-Channel 1
        Local Configuration:
                P-mLACP Enabled:      Yes
                Redundancy Group:     1
                Revertive Mode:       Revertive
                Primary VLANs:        20
                Secondary VLANs:      40
        Local Interface State:
                Interface ID: 1
                Port State:           Up
                Primary VLAN State:   Active
                Secondary VLAN State: Standby
        Peer Interface State:
                Interface ID: 1
                Primary VLAN State:   Active
                Secondary VLAN State: Standby
```

Step 7 **show nve vni 11111 detail**

Example:

```
IInterface  VNI        Multicast-group  VNI state
nve1       11111      N/A              Up
VNI Detailed statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
1682112875 107655224000 1681321674 107604587136
```

Step 8 **show lacp multi load group**

Example:

```
Interchassis Redundancy Group 1

                RG State:       Synchronized
                ICCP Version:   0
        Backbone Uplink Status: Connected
        Local Configuration:
                Node-id:        0

        Peer Information:
                State:          Up
                Node-id:        1
                ICCP Version:   0

States:      Active     - ACT           Standby    - SBY
             Down       - DN            AdminDown  - ADN
             Unknown    - UN            Reverting  - REV

P-mLACP Interfaces
Interface    Port State     Local VLAN State      Peer VLAN State
   ID          Local       Primary/Secondary     Primary/Secondary
   1             UP             ACT/SBY               ACT/SBY
```

Configuration Examples for VXLAN-MCLAG Active-Active High Availability Support

Example: Configuring VXLAN HA on Multicast Mode

The following example shows how to configure the VXLAN-MCLAG Active-Active High Availability Support feature on a multicast mode with two points of attachments (POA) connected to branch devices. The following is the configuration on the first POA—POA1.

```
ip multicast-routing distributed
ip pim bidir-enable
ip pim rp-address 4.4.4.4 bidir

redundancy
 mode sso
 interchassis group 1
  monitor peer bfd
  member ip 9.9.9.9
  backbone interface GigabitEthernet0/1/0
  mlacp system-priority 200
  mlacp node-id 0

bridge-domain 20
 member vni 7777
 member Port-channel1 service-instance 20
!

bridge-domain 40
 member vni 8888
 member Port-channel1 service-instance 40
!
interface Loopback10
 ip address 77.1.1.1 255.255.255.255
 ip pim sparse-dense-mode
!
interface Loopback11
 ip address 88.1.1.1 255.255.255.255
 ip pim sparse-dense-mode
!
interface Port-channel1
 no ip address
 negotiation auto
 lacp fast-switchover
 mlacp interchassis group 1
 mlacp mode active-active
 mlacp load-balance primary vlan 40
 mlacp load-balance secondary vlan 20
 service instance 20 ethernet
  encapsulation dot1q 20
 !
 service instance 40 ethernet
  encapsulation dot1q 40
 !
!
interface nve1
 no ip address
 member vni 7777 mcast-group 225.1.1.1
 source-interface Loopback10
!
interface nve2
 no ip address
 member vni 8888 mcast-group 226.1.1.1
 source-interface Loopback11
!

interface GigabitEthernet0/1/0
 ip address 192.168.20.1 255.255.255.0
 ip pim sparse-dense-mode
 negotiation auto
!

router ospf 10
 router-id 3.3.3.3
 network 0.0.0.0 255.255.255.255 area 10
!
```

The following is the configuration on the second POA—POA2.

```
ip multicast-routing distributed
ip pim bidir-enable
ip pim rp-address 4.4.4.4 bidir

redundancy
 mode sso
 interchassis group 1
  monitor peer bfd
  member ip 3.3.3.3
  backbone interface GigabitEthernet0/0/1
  mlacp system-priority 200
  mlacp node-id 1

bridge-domain 20
 member vni 7777
 member Port-channel1 service-instance 20
!

bridge-domain 40
 member vni 8888
 member Port-channel1 service-instance 40
!

interface Loopback10
 ip address 77.1.1.1 255.255.255.255
 ip pim sparse-dense-mode
!
interface Loopback11
 ip address 88.1.1.1 255.255.255.255
 ip pim sparse-dense-mode
!
interface Port-channel1
 no ip address
 negotiation auto
 no keepalive
 lacp fast-switchover
 mlacp interchassis group 1
 mlacp mode active-active
 mlacp load-balance primary vlan 20
 mlacp load-balance secondary vlan 40
 service instance 20 ethernet
  encapsulation dot1q 20
 !
 service instance 40 ethernet
```

```
  encapsulation dot1q 40
 !
!
interface nve1
 no ip address
 member vni 7777 mcast-group 225.1.1.1
 source-interface Loopback10
!
interface nve2
 no ip address
 member vni 8888 mcast-group 226.1.1.1
 source-interface Loopback11
!

interface GigabitEthernet0/1/0
 ip address 192.168.20.1 255.255.255.0
 ip pim sparse-dense-mode
 negotiation auto
!

interface GigabitEthernet0/0/1
 ip address 192.168.4.1 255.255.255.0
 ip pim sparse-dense-mode
 negotiation auto
end

router ospf 10
 router-id 9.9.9.9
 network 0.0.0.0 255.255.255.255 area 10
!
```

The following is the configuration on the first branch—Branch1.

```
ip multicast-routing distributed
ip pim bidir-enable
ip pim rp-address 4.4.4.4 bidir
!
bridge-domain 20
 member vni 7777
 member GigabitEthernet0/0/0 service-instance 20
!
interface Loopback10
 ip address 99.1.1.1 255.255.255.255
 ip pim sparse-dense-mode
!
interface nve1
 no ip address
 member vni 7777 mcast-group 225.1.1.1
 source-interface Loopback10
!
interface GigabitEthernet0/0/0
no ip address
negotiation auto
 service instance 20 ethernet
  encapsulation dot1q 20
 !
!
interface GigabitEthernet0/0/0
 ip address 192.168.3.1 255.255.255.0
 ip pim sparse-dense-mode
!
router ospf 10
 network 0.0.0.0 255.255.255.255 area 10
!
```

The following is the configuration on the second branch—Branch2.

```
ip multicast-routing distributed
ip pim bidir-enable
ip pim rp-address 4.4.4.4 bidir
!
bridge-domain 40
 member vni 8888
 member GigabitEthernet0/0/0 service-instance 40
!
interface Loopback11
 ip address 100.1.1.1 255.255.255.255
 ip pim sparse-dense-mode
!
interface nve1
 no ip address
 member vni 8888 mcast-group 226.1.1.1
 source-interface Loopback11
!
interface GigabitEthernet0/0/0
no ip address
 negotiation auto
 service instance 40 ethernet
  encapsulation dot1q 40
 !
!
interface GigabitEthernet0/0/1
 ip address 192.168.21.1 255.255.255.0
 ip pim sparse-dense-mode
negotiation auto
!
router ospf 10
network 0.0.0.0 255.255.255.255 area 10
!
```

Example: Configuring VXLAN HA on Unicast Mode

The following example shows how to configure the VXLAN-MCLAG Active-Active High Availability Support feature on an unicast mode with two points of attachments (POA) connected to branch devices. The following is the configuration on the first POA—POA1.

```
redundancy
 mode sso
 interchassis group 1
  monitor peer bfd
  member ip 9.9.9.9
  backbone interface GigabitEthernet0/1/0
  mlacp system-priority 200
  mlacp node-id 0

bridge-domain 20
 member vni 7777
 member Port-channel1 service-instance 20
!

bridge-domain 40
 member vni 8888
 member Port-channel1 service-instance 40
!
interface Loopback10
 ip address 77.1.1.1 255.255.255.255
```

```
!
interface Loopback11
 ip address 88.1.1.1 255.255.255.255
!
interface Port-channel1
 no ip address
 negotiation auto
 lacp fast-switchover
 mlacp interchassis group 1
 mlacp mode active-active
 mlacp load-balance primary vlan 40
 mlacp load-balance secondary vlan 20
 service instance 20 ethernet
  encapsulation dot1q 20
 !
 service instance 40 ethernet
  encapsulation dot1q 40
 !
!
interface nve1
 no ip address
 member vni 7777
  ingress-replication 99.1.1.1
 !
 source-interface Loopback10
!
interface nve2
 no ip address
 member vni 8888
  ingress-replication 100.1.1.1
 !
 source-interface Loopback11
!

router ospf 10
 router-id 3.3.3.3
 network 0.0.0.0 255.255.255.255 area 10
!
```

The following is the configuration on the second POA—POA2.

```
redundancy
 mode sso
 interchassis group 1
  monitor peer bfd
  member ip 3.3.3.3
  backbone interface GigabitEthernet0/0/1
  mlacp system-priority 200
  mlacp node-id 1

bridge-domain 20
 member vni 7777
 member Port-channel1 service-instance 20
!

bridge-domain 40
 member vni 8888
 member Port-channel1 service-instance 40
!

interface Loopback10
 ip address 77.1.1.1 255.255.255.255
!
interface Loopback11
 ip address 88.1.1.1 255.255.255.255
```

```
!
interface Port-channel1
 no ip address
 negotiation auto
 no keepalive
 lacp fast-switchover
 mlacp interchassis group 1
 mlacp mode active-active
 mlacp load-balance primary vlan 20
 mlacp load-balance secondary vlan 40
 service instance 20 ethernet
  encapsulation dot1q 20
 !
 service instance 40 ethernet
  encapsulation dot1q 40
 !
!
interface nve1
 no ip address
 member vni 7777
  ingress-replication 99.1.1.1
 !
 source-interface Loopback10
!
interface nve2
 no ip address
 member vni 8888
  ingress-replication 100.1.1.1
 !
 source-interface Loopback11
!

router ospf 10
 router-id 9.9.9.9
 network 0.0.0.0 255.255.255.255 area 10
!
```

The following is the configuration on the first branch—Branch1.

```
bridge-domain 20
 member vni 7777
 member GigabitEthernet0/0/0 service-instance 20
!
interface Loopback10
 ip address 99.1.1.1 255.255.255.255
!
interface nve1
 no ip address
 member vni 7777
     ingress-replication 77.1.1.1
 source-interface Loopback10
!
interface GigabitEthernet0/0/0
no ip address
negotiation auto
 service instance 20 ethernet
  encapsulation dot1q 20
 !
!
router ospf 10
 network 0.0.0.0 255.255.255.255 area 10
!
```

The following is the configuration on the second branch—Branch2.

```
bridge-domain 40
 member vni 8888
 member GigabitEthernet0/0/0 service-instance 40
!
interface Loopback11
 ip address 100.1.1.1 255.255.255.255
!
interface nve1
 no ip address
 member vni 8888
     ingress-replication 88.1.1.1
 source-interface Loopback11
!
interface GigabitEthernet0/0/0
no ip address
negotiation auto
 service instance 40 ethernet
  encapsulation dot1q 40
 !
!
router ospf 10
 network 0.0.0.0 255.255.255.255 area 10
!
```

Additional References for VXLAN-MCLAG Active-Active High Availability Support

Related Documents

Related Topic	Document Title
Carrier Ethernet commands	Cisco IOS Carrier Ethernet Command Reference

Technical Assistance

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

Feature Information for VXLAN-MCLAG Active-Active High Availability Support

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

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

Table 32: Feature Information for VXLAN-MCLAG Active-Active High Availability Support

Feature Name	Releases	Feature Information
VXLAN-MCLAG Active-Active High Availability Support	Cisco IOS XE 3.16S	The VXLAN-MCLAG Active-Active High Availability Support feature implements dual-home device with pseudo Multichassis Link Aggregation Control Protocol (pMLACP) redundancy mode and layer 2 VXLAN on the Cisco ASR 1000 Series Aggregation Services Routers. The following commands were introduced by this feature: **show lacp internal, show nve interface nve1, show nve peersshow platform software ethernet fp ac bridge-domain binding, show bridge-domain 20, show lacp multi-chassis load-balance port-channel, show nve vni 11111 detail, show lacp multi load group**

CHAPTER 23

VxLAN Support

This module contains information about VxLAN (Virtual eXtensible Local Area Network) Layer 2 gateway feature support on the Cisco ASR 1000 Series Routers. VxLAN is a technology that provides a Layer 2 overlay network, allowing for network isolation. The standard 802.1q VLAN implementation limits the number of tags to 4096. However, cloud service providers may want to operate more than 4096 virtual networks. VxLAN uses a 24-bit network ID, which allows for a much larger number of individual i networks to be operated.

Finding Feature Information

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

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

Prerequisites for VxLAN Support

The following are the prerequisites to configuring the Cisco ASR 1000 Routers as a VxLAN Layer 2 gateway:

1. Configure the loopback interface.
2. Configure the IP unicast reachability to remote VTEP's.
3. Configure Bidirectional Protocol Independent Multicast (PIM) or Protocol Independent Multicast-Sparse Mode (PIM-SM).

For more information, see the IP Multicast: PIM Configuration Guide, Cisco IOS XE Release 3S .

Information About VxLAN Support

This feature enables the Cisco ASR 1000 Series Routers to act as a Layer 2 VxLAN gateway to provide support to bridge traffic across VxLAN segments in a hypervisor and on VLANs on physical servers. The operation of a VxLAN Layer 2 gateway is based on the data plane MAC address learning and flooding of multidestination traffic (such as unknown unicast, multicast, or broadcast frames) using IP multicast.

Acting as a VxLAN Layer 2 gateway, the Cisco ASR 1000 Routers can send and receive packets on multiple VxLAN networks, and provide connectivity between the hosts in a VLAN network and the virtual machines operating on a VxLAN network.

A VXLAN supports different modes for flood traffic:

- Multicast Mode—A VXLAN uses an IP multicast network to send broadcast, multicast, and unknown unicast flood frames. Each multicast mode VXLAN has an assigned multicast group IP address. When a new VM joins a host in a multicast mode VXLAN, the Virtual Tunnel Endpoint (VTEP) joins the assigned multicast group IP address by sending IGMP join messages. Flood traffic, broadcast, multicast and unknown unicast from the VM is encapsulated and is sent using the assigned multicast group IP address as the destination IP address. Packets sent to known unicast MAC addresses are encapsulated and sent directly to the destination server Virtual Tunnel Endpoint (VTEP) IP addresses.
- Unicast-Only Mode—A VXLAN uses each VEM's single unicast IP address as the destination IP address to send broadcast, multicast, and unknown unicast flood frames of the designated VTEP on each VEM that has at least one VM in the corresponding VXLAN. When a new VM joins the host in a unicast-mode VXLAN, a designated VTEP is selected for receiving flood traffic on that host. This designated VTEP is communicated to all other hosts through the Virtual Supervisor Module (VSM). Flood traffic (broadcast, multicast, and unknown unicast) is replicated on each VEM's designated VTEP in that VXLAN by encapsulating it with a VXLAN header. Packets are sent only to VEMs with a VM in that VXLAN. Packets that have a unicast MAC address are encapsulated and sent directly to the destination server's VTEP IP address.
- MAC Distribution Mode (supported only in unicast mode)—In this mode, unknown unicast flooding in the network is eliminated. The VSM learns all the MAC addresses from the VEMs in all the VXLANs and distributes those MAC addresses with VTEP IP mappings to other VEMs. Therefore, no unknown unicast MAC address exists in the network when the VMs on the VEMs are communicating and controlled by the same VSM.

The VxLAN Layer 2 gateway performs the following functions:

- Provides support to bridge traffic between a host in a VLAN domain and VMs behind a virtual switch (vSwitch) in a VxLAN domain. The VLAN and the virtual network identifier (VNI) on the VxLAN should be configured as member ports in the same bridge domain.
- Implements the Virtual Tunnel Endpoint (VTEP) function, which encapsulates the Layer 2 packet on the IP/UDP tunnel with the VxLAN header (VNI) information before sending it to a multicast group or particular virtual switch on the VxLAN domain.
- The VTEP function removes the VxLAN header, identifies the bridge domain under which the VNI is configured and then bridges the inner L2 packet to the VLAN side. The bridge function also learns the remote MAC address (the VM's MAC address behind the virtual switch).
- The Layer 2 gateway carries the inner payload of non-IP (Layer 2 traffic), IPv4, and IPv6 traffic over the VxLAN VNI member.

Limitations of VxLAN Support

1. Platforms that support a new scale number (8192 or 16000) require an 8G RP memory. Scale number for RP memory that is less than 8G is unchanged.
2. Scale number on platform RP+ESP5 and ASR1002F is unchanged.
3. VxLAN is not supported on ISR4000 series platforms before Cisco IOS XE Everest 16.5.1.
4. The maximum NVE interface number is unchanged on all platforms.
5. The NVE source is supported for lookback interface before Cisco IOS XE Denali 16.3. After Cisco IOS XE Denali 16.3, it can support physical interfaces as well.
6. The scale enhancement is applicable only for the VxLAN layer 2 and layer 3 gateway feature. Other bridge-domain related features are not impacted.
7. RP switchover for VxLAN is not supported on these platforms before Cisco IOS XE Denali 16.3.
8. Only one VNI ID on every bridge-domain is supported.

New Scale Number after Enhancements

The following table lists new VxLAN scale numbers on different platforms after enhancements. All platforms that support a new scale number (8192 or 16000) require an 8G RP memory.

Platform	MAX BD per system	MAX BDI interface per system	MAX VNI per system
RP+ESP200	16000	16000	16000
RP+ESP100	16000	16000	16000
RP+ESP40	16000	16000	16000
RP+ESP20	16000	16000	16000
RP+ESP10	16000	16000	16000
ASR1002-X	16000	16000	16000
ASR1001-X	16000	16000	16000
ASR 1001	8192	8192	8192
CSR1000v	8192	8192	8192

Configuring VxLAN Layer 2 Gateway with Multicast

Configuring the VxLAN UDP Destination Port (Optional)

The default VxLAN UDP destination is 4789. If you want to change the VxLAN UDP destination port value, you must change it before configuring the network virtualization endpoint (NVE) interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vxlan udp port** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**vxlan udp port** *number* **Example:** `Router(config)# vxlan udp port 1000`	Configures the VxLAN UDP destination port number. The default value is 4789.

Creating the Network Virtualization Endpoint (NVE) Interface

You create the network virtualization endpoint (NVE) interface and then assign member virtual network identifiers (VNIs) to it. The mapping between the VNI range and the multicast group range is either one-to-one or many-to-one.

SUMMARY STEPS

1. **interface nve** *number*
2. **source-interface loopback** *number*
3. **member vni** {*range* | *startnumber-endnumber*} **multicast-group** *startip-address endip-address*
4. **member vni** *range*
5. **ingress-replication** *Unicast IP Addresses*
6. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**interface nve** *number* **Example:** `Router(config)# interface nve 1`	Creates a network virtualization endpoint (NVE) interface and enters NVE interface configuration mode.
Step 2	**source-interface loopback** *number* **Example:** `Router(config-if)# source-interface loopback 0`	Assigns the previously-created loopback interface to the NVE interface.
Step 3	**member vni** {*range* | *startnumber-endnumber*} **multicast-group** *startip-address endip-address* **Example:** `Router(config-if)#` **`member vni 7115 multicast-group 225.1.1.1`**	Creates a VNI member or a range of VNI members. Repeat this step for each VNI to be added to the NVE interface. The valid values for the VNI number are from 4096 to 16777215.
Step 4	**member vni** *range* **Example:** `Router(config-if)#` **`member vni 7115`**	Creates a VNI member or a range of VNI members. Repeat this step for each VNI to be added to the NVE interface. The valid values for the VNI number are from 4096 to 16777215.

Step 5	**ingress-replication** *Unicast IP Addresses*
	Example:
	`Router(config-if-nve-vni)#` **`ingress-replication 225.1.1.1`**
	`ingress-replication 225.1.1.2`
		Sets up ingress-replication unicast addresses which enables the headend replication functionality.

Step 6	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Enables the NVE interface.

Creating the Access Ethernet Flow Point (EFP)

After the member VNI is created, you must create the access Ethernet Flow Point (EFP) for the VLAN interface.

SUMMARY STEPS

1. **interface GigabitEthernet** *number*
2. service instance *id* ethernet
3. encapsulation dot1q *vlan-ID*

4. rewrite ingress tag pop 1 symmetric

DETAILED STEPS

	Command or Action	Purpose
Step 1	**interface GigabitEthernet** *number* **Example:** `Router(config)# interface GigabitEthernet1`	Enters interface configuration mode.
Step 2	service instance *id* ethernet **Example:** `Router(config-if)# service instance 20 ethernet`	Configures an Ethernet service instance on the overlay interface being configured and enters service instance configuration mode. • The service instance identifier range is from 1 to 8000.
Step 3	encapsulation dot1q *vlan-ID* **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Defines the VLAN encapsulation format as IEEE 802.1Q and specifies the VLAN identifier.
Step 4	rewrite ingress tag pop 1 symmetric **Example:** `Router(config-if-srv)# rewrite ingress tag pop 1 symmetric`	Removes the VLAN tag in the Layer 2 traffic before switching to the outgoing VxLAN interface. **Note** This command is required to remove the VLAN tag before sending the VLAN traffic to VxLAN and adding the VLAN tag in the reverse direction.

Mapping the VLAN to the Bridge Domain

You must map the VLAN created in the previous procedure to the bridge domain.

SUMMARY STEPS

1. **bridge-domain** *bridge-id*
2. **member** *interface* **service-instance** *id*
3. **member vni** *vni-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**bridge-domain** *bridge-id* **Example:** `Router(config)# bridge-domain 10`	Creates a bridge domain and enters bridge domain configuration mode. The valid range for bridge-id is 1-4096.
Step 2	**member** *interface* **service-instance** *id* **Example:**	Binds the bridge domain to the service instance.

	Command or Action	Purpose
	`Router(config-bdomain)# member gigabitEthernet 1 service-instance 1`	
Step 3	**member vni** *vni-id* **Example:** `Router(config-bdomain)# member vni 1010`	Maps the VNI to the bridge domain.

What to do next

The following example displays the NVE VNIs configured on the router:

```
Router# show nve vni

Interface  VNI          mcast       VNI state
nve1       5000         230.1.1.1       UP        L2DP 2 N/A
```

The following example displays the NVE VNIs assigned to NVE interface 1:

```
Router(config)# show nve vni interface nve1
Interface  VNI          mcast       VNI state
nve1       5000         230.1.1.1       UP        L2DP 2 N/A
```

The following example shows the status of NVE interface 1:

```
Router(config)# show nve interface nve1
Interface: nve1, State: Admin Up, Oper Up Encapsulation: Vxlan
source-interface: Loopback0 (primary:11.11.11.11 vrf:0)
```

The following example shows a detailed display for NVE interface 1:

```
Router(config)# show nve interface nve1 detail
Interface: nve1, State: Admin Up, Oper Up Encapsulation: Vxlan
source-interface: Loopback0 (primary:11.11.11.11 vrf:0)
Pkts In   Bytes In   Pkts Out  Bytes Out
0          0          0          0
```

The following example shows the NVE peers configured on the router:

```
Router(config)# show nve peers
Interface Peer-IP         VNI      Up Time
nve1        230.1.1.1     5000       UP        L2DP 2 N/A
nve2       1.1.1.3        2030      20h
```

The following example shows the bridge domain configuration with the entry in bold displaying the VM's MAC address that was learned on the VxLAN VNI:

```
Router# show bridge-domain 1000
Bridge-domain 1000 (3 ports in all)
State: UP                    Mac learning: Enabled
Aging-Timer: 300 second(s)
    GigabitEthernet1 service instance 1000
    GigabitEthernet3 service instance 1000
    vni 7639335
   MAC address    Policy  Tag      Age  Pseudoport
   0050.56A4.ECD2 forward dynamic  297  nve1.VNI7639335 VxLAN
src:10.0.0.1  dst:10.0.0.2
   0050.56A4.257A forward dynamic  297  GigabitEthernet3.EFP1000
```

Configuring VxLAN Layer 2 Gateway with Unicast

The following example shows VxLAN with unicast ingress-replication which is a point-to-point (unicast) configuration.

```
interface Loopback0
ip address 11.11.11.11 255.255.255.255
!
interface nve1
no ip address
member vni 5001
  ingress-replication 22.22.22.22  < Remote L2 GW loopback ip>
!
source-interface Loopback0
!
bridge-domain 1
member vni 5001
member GigabitEthernet0/2/0 service-instance 1
interface GigabitEthernet0/2/0
service instance 1 ethernet
encapsulation dot1q 100
rewrite ingress tag pop 1 symmetric
```

Feature Information for VxLAN Support

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

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

Table 33: Feature Information for VxLAN Support

Feature Name	Releases	Feature Configuration Information
VxLAN Support	Cisco IOS XE Release 3.13.1S	This feature was introduced on the Cisco ASR 1000 Series Routers.
	Cisco IOS XE Fuji 16.9	This feature was introduced on the following: • Cisco ISR 1000 Series Integrated Services Routers. • Cisco ISR 4000 Series Integrated Services Routers.
Protocol Independent Multicast-Sparse Mode (PIM-SM) Support	Cisco IOS XE Release 3.17S	This feature was introduced on the Cisco ASR 1000 Series Routers. No commands were introduced or modified for this feature.

Feature Name	Releases	Feature Configuration Information
Support for multiple ingress replication peers	Cisco IOS XE Everest 16.5.1b	The VXLAN feature was modified to support multiple ingress replication peers on the Cisco ASR 1000 Series Routers. The **ingress-replication** command was modified to support multiple replication peers for every VNI up to 32 nodes.

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 24

VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

VxLAN (Virtual eXtensible Local Area Network) GPE (Generic Protocol Extension) is intended to extend the existing VxLAN protocol to provide protocol typing, OAM, and versioning capabilities.

Restrictions for VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

- VxLAN GPE tunnel and VxLAN dummy-L2 tunnel do not support IS-IS.
- Layer2 traffic is not supported.
- IPv6 for underlay encapsulation is not supported.

Information About VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

Overview

Virtual eXtensible Local Area Network (VxLAN) defines an encapsulation format that encapsulates Ethernet frames in an outer UDP/IP transport. As data centers evolve, the need to carry other protocols encapsulated in an IP packet is required, and the need to provide increased visibility and diagnostic capabilities within the overlay. The VxLAN header does not specify the protocol being encapsulated and therefore is currently limited to encapsulating only Ethernet frame payload, nor does it provide the ability to define Operations, Administration, and Maintenance (OAM) protocols. In addition, new transports need not use transport layer port numbers to identify tunnel payload, rather it encourages encapsulations to use their own identifiers for this purpose. VxLAN GPE (Generic Protocol Extension) is intended to extend the existing VxLAN protocol to provide protocol typing, OAM and versioning capabilities.

The following are the main features of VxLAN GPE Tunnel:

- Encapsulates layer-3 packets directly into a VxLAN tunnel without any layer-2 bridge-domain requirements or dependencies.
- Provides an equal-cost multi-path (ECMP) entropy benefits on the underlay (core) network by calculating the outer-source UDP port based on the inner IP protocol, source or destination IP addresses and L4 port numbers (5-tuple).
- Leverages VxLAN-GPE draft-IETF, which provides direct upper layer protocols options such as, IPv4, IPv6, Ethernet (MAC), Network-Service-Header (NSH) without a layer-2 header.
- Overlay encapsulation is supported for both IPv4 and IPv6, whereas underlay encapsulation is supported only for IPv4.
- 8K Tunnel interfaces with VxLAN GPE or VxLAN dummy-L2 mode are supported.
- 3-tuple hash is enabled to generate UDP source port for both VxLAN GPE and VxLAN Dummy-L2 tunnels for all the packets.
- UDP source port hash uses 3 tuples for fragments, and 5 tuples for non-fragments.

Feature Information for VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

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

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

Table 34: Feature Information for VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

Feature Name	Releases	Feature Information
VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel	Cisco IOS XE Denali 16.3.1	The following commands were introduced by this feature: **vxlan gpe-tunnel-udp-port xxx**.
Manually configure the source port range for VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel	Cisco IOS XE Everest 16.5.1b	The following commands were introduced by this feature: • **vxlan source-port-range udp** • **vxlan source-port-range gpe-tunnel-udp** • **vxlan source-port-range dummy-l2-tunnel-udp**
Microsoft - VxLAN GPE Tunnel IPv6 Support VxLAN 8K GPE Tunnel Support	Cisco IOS XE Fuji 16.7.1	Overlay encapsulation is supported for both IPv4 and IPv6, whereas underlay encapsulation is supported only for IPv4. 8K Tunnel interfaces with VxLAN GPE or VxLAN dummy-L2 mode are supported.

Feature Name	Releases	Feature Information
VXLAN Fragment UDP Source Port	Cisco IOS XE 16.8.1	Global tunnel CLIs are added to enable 3-tuple hash to generate UDP source port for both VxLAN-GPE and VxLAN Dummy-L2 tunnels. Support both global and under interface tunnel.
IPv6 VxLAN GPE Tunnel and IPv6 VxLAN Dummy-L2 Tunnel	Cisco IOS XE Gibraltar 16.12.1	VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel support IPv6 tunnel mode.

How to Configure VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

Configuring VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

1. Configure Vxlan GPE Tunnel on VTEP1.

```
interface Tunnel1
  ip address 192.168.1.1 255.255.255.0
  tunnel source GigabitEthernet2
  tunnel mode vxlan-gpe ipv4
  tunnel destination 20.1.1.17
  tunnel vxlan vni 123456
```

2. Configure Vxlan GPE Tunnel on VTEP2.

```
interface Tunnel1
  ip address 192.168.1.2 255.255.255.0
  tunnel source GigabitEthernet2
  tunnel mode vxlan-gpe ipv4
  tunnel destination 20.1.1.16
  tunnel vxlan vni 123456
```

3. Configure Vxlan Dummy-L2 Tunnel on VTEP1.

```
interface Tunnel0
  ip address 192.168.2.3 255.255.255.0
  tunnel source GigabitEthernet2
  tunnel mode vxlan ipv4 default-mac
  tunnel destination 20.1.1.17
  tunnel vxlan vni 123456
```

4. Configure Vxlan Dummy-L2 Tunnel on VTEP2.

```
interface Tunnel1
  ip address 192.168.2.1 255.255.255.0
  tunnel source GigabitEthernet2
  tunnel mode vxlan ipv4 default-mac
  tunnel destination 20.1.1.16
  tunnel vxlan vni 123456
```

5. (Optional) Change UDP dst port for Vxlan Dummy-L2 Tunnel. Default UDP port of Dummy-L2 Tunnel is 4789.

```
(config)#vxlan dummy-l2-tunnel-udp-port 4789
```

6. (Optional) Change UDP dst port for Vxlan GPE Tunnel. Default UDP port of GPE Tunnel is 4790.

```
(config)#vxlan gpe-tunnel-udp-port 4790
```

Verifying VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

```
##show platform software vxlan fp active udp-port
VXLAN UDP Port: 6000
VXLAN GPE Tunnel UDP Port: 4000
VXLAN Dummy L2 Tunnel UDP Port: 6000
VXLAN UDP Source Port Range: 1025 - 65535
VXLAN GPE Tunnel UDP Source Port Range: 1025 - 65535
VXLAN Dummy L2 Tunnel UDP Source Port Range: 1025 – 65535
VXLAN GPE Tunnel UDP Source Port Hash: 3 tuples
VXLAN  Dummy L2 Tunnel UDP Source Port Hash: 3 tuples
```

Perform the following steps to add new tunnel mode:

```
#interface tunnel0
     tunnel mode vxlan ipv4 default-mac | xxxx.xxxx.xxxx xxxx.xxxx.xxxx


#interface tunnel1
    tunnel mode vxlan-gpe ipv4
```

Perform the following steps to add VxLAN VNI in tunnel:

```
#interface tunnel0
tunnel vxlan vni xxxx
```

Perform the following steps to add VxLAN source port hash in tunnel:

```
#interface tunnel0
vxlan source-port-hash 3-tuple
vxlan source-port-hash 3-tuple-for-fragments
```

Configuration Examples for VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

Example: VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

Example: VxLAN GPE Tunnel

```
#interface Tunnel0
 ip address 192.168.2.1 255.255.255.0
 ipv6 address 2001::1/64
 tunnel source GigabitEthernet2
 tunnel mode vxlan ipv4 default-mac
 tunnel destination 20.1.1.16
 tunnel vxlan vni 123456
```

```
#interface Tunnel1
 ip address 192.168.1.2 255.255.255.0
 ipv6 address 2002::2/64
 tunnel source GigabitEthernet2
 tunnel mode vxlan-gpe ipv4
 tunnel destination 20.1.1.16
 tunnel vxlan vni 123456
```

```
#sh pl soft vxlan f0 udp-port
VXLAN UDP Port: 4789
VXLAN GPE Tunnel UDP Port: 4790
VXLAN Dummy L2 Tunnel UDP Port: 4789
VXLAN UDP Source Port Range: 600 - 6000
VXLAN GPE Tunnel UDP Source Port Range: 400 - 4000
VXLAN Dummy L2 Tunnel UDP Source Port Range: 1025 - 65535
VXLAN GPE Tunnel UDP Source Port Hash: 5 tuples
VXLAN  Dummy L2 Tunnel UDP Source Port Hash: 5 tuples
```

Additional References for VxLAN GPE Tunnel and VxLAN Dummy-L2 Tunnel

Standards and RFCs

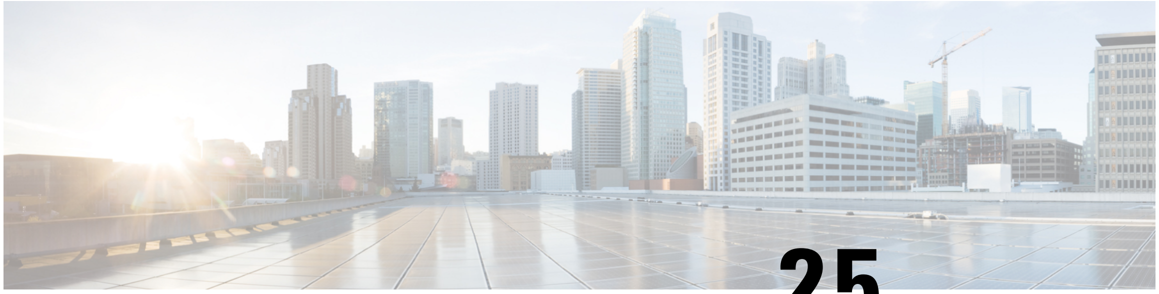
Standard/RFC	Title
draft-ietf-nvo3-vxlan-gpe-02.txt	*Generic Protocol Extension for VXLAN*

MIBs

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

Technical Assistance

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



CHAPTER 25

EVPN VxLAN L3

This chapter provides information on Layer 3 Data-Center-Interconnect (DCI) VXLAN EVPN Support.

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 EVPN VxLAN L3

- VNI range CLI for L3VNI is not supported.
- Egress traffic stops, if local VNI is down.
- L3 VNI and L2 VNI co-existence in the same bridge domain as L3 VNI is not supported.
- MAC learning is not done with L3VNI via control plane learning.
- IPV6 overlay and underlay are not supported.

Information About EVPN VxLAN L3

Data Center Interconnect VXLAN Layer 3 Gateway

The Cisco device can serve as a Data Center Interconnect (DCI) L3 Gateway to provide IP connectivity between multi-tenant remote Data Center sites. The multi-tenant Data Centers use VxLAN encapsulation to carry separate tenant IP traffic. The VXLAN-enabled Data Center sites use MP-BGP EVPN control plane for distributing both Layer-2 and Layer-3 forwarding information within the site. RFC 5512 and draft-ietf-bess-evpn-inter-subnet-forwarding-00 define how MP-BGP Network Layer Reachability Information (NLRI) carries VXLAN encapsulation as well as L2/L3 forwarding information details to provide an integrated routing and bridging solution within the Data Center site.

Route Targets

For each VRF on the DCI router, there are two sets of manually configured import and export route-targets. One set of import and export route-targets is associated with the Data Center BGP neighbor that uses EVPN address-family to exchange L3 information; the other set of import and export route-targets is associated with the L3VPN BGP neighbor that use VPNv4 unicast address-family to exchange L3 information. This separation of route targets (RTs) enables the two sets of RTs to be independently configured. The DCI router effectively stitches the two set of RTs. The RTs associated with the EVPN BGP neighbor are labelled as stitching RTs. The RTs associated with the L3VPN BGP neighbor are normal RTs.

A new keyword is added to the existing route-target configuration to specify the route targets to be used when doing EVPN-VXLAN related processing. The base (existing) route target configuration does not affect EVPN-VXLAN related processing. You can have the same RT values for both base and VxLAN routes.

Local VPNv4 Routes Advertisement

On the DCI router, the locally sourced VPNv4 routes can be advertised to the BGP EVPN neighbors with the normal route targets (RTs) configured for the VRF or the stitching RTs associated with the BGP EVPN neighbors. By default, these routes are advertised with the normal route targets.

Note You cannot configure the advertise command for VPNv4 or VPNv6 neighbors. RTs can be applied only to the sourced routes and routes learned from VRF neighbors.

Data Center VXLAN with Support for MP-BGP

The Data Center VXLAN uses MP-BGP for control-plane learning of end-host Layer 2 and Layer 3 reachability information. The DCI router is configured with a VXLAN Tunnel EndPoint (VTEP). You also need to run the host-reachabilty protocol BGP command to specify that control-plane learning within Data center site is through BGP routing protocol.

The DCI Gateway router and the EVPN BGP neighbor (Data Center BGP neighbor) exchange BGP EVPN NLRIs of route type 5 that carry L3 routing information and associated VXLAN encapsulation information.

EVPN Route Targets

A new keyword is added to the existing route-target configuration to specify the route targets to be used when doing EVPN-VXLAN related processing. The base (existing) route target configuration does not affect EVPN-VXLAN related processing. You can have the same RT values for both base and vxlan routes

MAC/IP Advertisement Route and IP Prefix Route is supported. The l2vpn evpn address-family can be configured and neighbors can exchange EVPN NLRI. The l2vpn-evpn-prefix-advertisement is supported fully and for the non-MAC portions only the NLRI is supported. IP Prefix route type is added to carry IP prefixes. The IP Prefix NLRI can carry IPv4 Prefix or IPv6 Prefix. The NLRI length determines whether it has IPv4 Prefix or IPv6 Prefix.

NLRI Format:

```
[Type][Len][RD][ESI][ETag][IP Addr Mask][IP Addr][GW IP Addr][Label]
```

Key:

```
[Type][ETag][IP Addr Len][IP Addr]
```

When BGP attribute, encapsulation type EXTCOMM value of 0x8 (VxLAN) is present, then Label carries VNI (VXLAN ID).

EVPN RT5 and RT2 that contain a RT matching an import “stitching RT” specified in a vrf configuration is accepted by the router and imported into the corresponding BGP L3VPN vrf. The resulting L3VPN prefix retains the same route target. L3VPN routes that are imported into EVPN via “advertise l2vpn evpn” contains route targets specified by that vrf’s export “stitching RT”. Any original route targets is removed.

Bridge Domain Interface

Bridge Domain Interface (BDI) is used for Inter-VLAN routing for EVC. It supports ping from local BDI interface to peer BDI/BVI/SVI. ARP is not used to resolve adjacency. BGP is asked to advertise the BDI IP address in EVPN route and use RMAC as an adjacency.

Downstream VNI

A downstream VNI is assigned at the downstream BGP peer. The BGP peer sends VNI as part of EVPN route type 2 or 5, so that it can use the VNI to send EVPN traffic to peer. This VNI is called as egress VNI; this egress VNI is used to send EVPN traffic to peer on data path. BGP also sends the local VNI to peer as part of EVPN route type 2 or 5 and it is expected from the peer to send EVPN traffic with the VNI, so that it can route the PKT to right VRF. This VNI is called as ingress VNI.

For the local VNI, VNI number range is 4k to 16m. For the egress VNI, valid VNI number range can be any valid VNI number, from 1-16m.

Router MAC

EVPN introduces a Router’s MAC extended community to exchange Router’s MAC between EVPN peer. BGP send BDI’s MAC address to EVPN Peer as its RMAC. By default, all the BDI interface share the same MAC address, so all EVPN VRF will send the same RMAC to EVPN peer by default. It is flexible to configure MAC address of BDI interface. So, it is possible that different EVPN VRF may send different RMAC to EVPN peer.

VRF Lite

VRF-lite (VPN routing/forwarding) allows a service provider to support two or more VPNs with overlapping IP addresses. VRF-lite is achieved by configuring sub-interfaces (VLANs) on a physical interface and by putting each sub-interface in a VRF.

EVPN Route Type 2 - MAC Advertisement Route

MAC Advertisement Route can be used to carry only MAC Address or MAC Address and IP Address (/32 for IPv4 or /128 for IPv6).

NLRI Format:

```
[Type][Len][RD][ESI][ETag][MAC Addr Mask][MAC Addr][IP Addr Len][IP Addr] [Label1] [Label2]
```

Key:

```
[Type][ETag][MAC Addr Len][MAC Addr][IP Addr Len][IP Addr]
[Type][ETag][MAC Addr Len][MAC Addr][IP Addr Len]
```

Label1 is associated with MAC Address and Label2 is associated with IP Address. When BGP attribute, encapsulation type EXTCOMM value of 0x8 (VxLAN) is present, then Label carries VNI (VXLAN ID).

L3 VRF EVPN Import

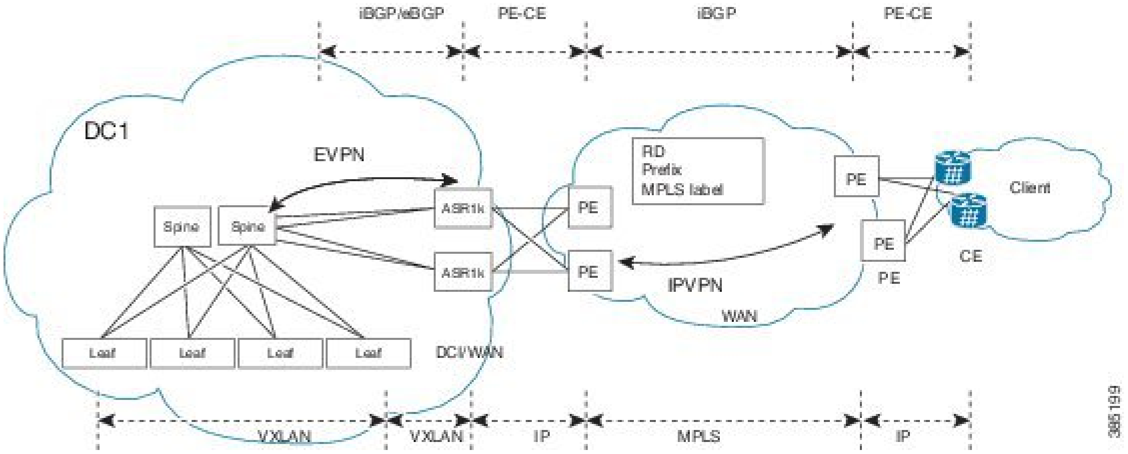
To advertise L3 VPN routing and forwarding (VRF) prefixes to EVPN neighbors define a new import type that takes prefixes from VRF neighbors, redistributed VRF routes, and import them into EVPN table. The import of VRF routes is controlled per VRF. The import of VRF is performed only when `advertise l2vpn evpn` is configured under that VRF and local VTEP is up.

EVPN DCI Solution

ASR1000 (IOS-XE Platform) series routers, acting as a Data Centre Interconnect (DCI) device can be deployed at the edge of two Cisco Data Center solutions, that is, Nexus 9000 Standalone-mode Data Centre or Nexus 9000 ACI-mode Data Centre. It provides flexible and safe WAN connections to the Internet or Branch sites with multiple different WAN types. Currently ASR1000 supports multiple WAN connection types, including iWAN, MPLS VPN(PE and ASBR), DMVPN, and VRF Lite. You can also deploy more than one ASR1000 router as multihoming deployment, if you require traffic load balancing, redundancy or customized path selection policy based on special requirements of different applications.

How to Configure EVPN VxLAN L3

The following is the sample topology that is used as an example to explain the configuration of this feature.



Configuring Customer Edge (CE) 1 Using VRF Lite

1. Define VRF and IPv4 address family. EVPN RT is 65535:1

```
vrf definition evpn1
 rd 65535:1
 address-family ipv4
route-target both 65535:1 stitching
exit-address-family
!
```

2. Define Bridge Domain and associate vxlan vni 3000.

```
bridge-domain 200
 member vni 30000
Interface loopback0
  ip address 33.33.33.33 255.255.255.255
```

3. Define Bridge Domain Interface (BDI).

```
interface BDI200
 vrf forwarding evpn1
 ip address 100.1.1.1 255.255.255.0
 encapsulation dot1Q 200
```

4. Create Interface NVE1.

```
Interface gi0/0/0.2
  enc dot1q 2
  ip address 4.0.0.1 255.255.255.0
Interface gi0/0/1.2
  enc dot1q 2
  vrf forwarding evpn1
  ip address 3.3.3.1 255.255.255.0
interface nve1
 no ip address
 source-interface Loopback0
 host-reachability protocol bgp
 member vni 30000 vrf evpn1
```

5. Define OSPF for underlay reachability.

```
Router ospf 100
  router-id 33.33.33.33
  network 33.33.33.33 0.0.0.0 area 0
  network 4.0.0.1 0.0.0.0 area 0
!
```

6. Define BGP and EVPN address-family.

```
router bgp 65535
 bgp router-id 33.33.33.33
 neighbor 44.44.44.44 remote-as 65535
 neighbor 44.44.44.44 update-source Loopback0
 !
 address-family l2vpn evpn
  neighbor 44.44.44.44 activate
  neighbor 44.44.44.44 send-community both
 exit-address-family
 !
 address-family ipv4 vrf evpn1
  advertise l2vpn evpn
  neighor 3.3.3.254 remote-as 65530
  neighor 3.3.3.254 update-source Gi0/0/1.2
  neighor 3.3.3.254 ebgp-multihop 255
  redistribute connected
 exit-address-family
```

Configuring Provider Edge 1

Define VRF and RD/RT.

```
vrf definition vrf1
 rd 65530:1
 address-family ipv4
  route-target both 65530:1
exit-address-family
!
interface loopback0
  ip address 33.33.33.22 255.255.255.255
Interface GigabitEthernet0/0/0.2
  enc dot1q 2
  vrf forwarding vrf1
  ip address 3.3.3.254 255.255.255.0
Interface gigabitEthernet0/0/1
  mpls ip
  ip address 2.2.2.1 255.255.255.0
!
Router ospf 100
  router-id 33.33.33.22
  network 33.33.33.22 0.0.0.0 area 0
  network 2.2.2.1 0.0.0.0 area 0
!

router bgp 65530
 bgp router-id 33.33.33.22
 neighbor 22.22.22.22 remote-as 65530
 neighbor 22.22.22.22 update-source Loopback0
 !
 address-family vpnv4
  neighbor 22.22.22.22 activate
  neighbor 22.22.22.22 send-community both
 exit-address-family
 !
 address-family ipv4 vrf vrf1
  neighor 3.3.3.253 remote-as 65535
  neighor 3.3.3.253 update-source Gi0/0/0.2
  neighor 3.3.3.253 ebgp-multihop 255
  redistribute connected
 exit-address-family
```

Configuring Provider Edge 2 and Branch Router

```
vrf definition vrf1
 rd 65530:1
 address-family ipv4
  route-target both 65530:1
exit-address-family
!
interface loopback0
  ip address 22.22.22.22 255.255.255.255
!
Interface GigabitEthernet0/0/0.200
  enc dot1q 200
  vrf forwarding vrf1
  ip address 1.1.1.254 255.255.255.0
!
Interface gigabitEthernet0/0/1
  mpls ip
  ip address 2.2.2.254 255.255.255.0
!
Router ospf 100
  router-id 22.22.22.22
  network 22.22.22.22 0.0.0.0 area 0
  network 2.2.2.254 0.0.0.0 area 0
!
router bgp 65530
 bgp router-id 22.22.22.22
 neighbor 33.33.33.22 remote-as 65530
 neighbor 33.33.33.22 update-source Loopback0
 !
 address-family vpnv4
  neighbor 33.33.33.22 activate
  neighbor 33.33.33.22 send-community both
 exit-address-family
 !
 address-family ipv4 vrf vrf1
   redistribute connected
 exit-address-family
```

Configuring Customer Edge 2

```
Interface GigabitEthernet0/0/0.200
  enc dot1q 200
ip address 1.1.1.1 255.255.255.0
ip route 0.0.0.0 0.0.0.0 1.1.1.254
```

Importing Between EVPN and VRF/VPN

```
router bgp 100
 address-family ipv4 vrf example-vrf
  advertise l2vpn evpn
  neighbor 7.7.7.7 remote-as 400
  neighbor 7.7.7.7 activate
 exit-address-family
```

Verifying EVPN VxLAN L3

Use the following commands to verify the configuration:

- **show ip bgp l2vpn evpn**: Displays Layer 2 Virtual Private Network (L2VPN) address family information from the Border Gateway Protocol (BGP) table.
- **show mlrib evpn mac**: Displays the MLRIB information pertaining to an EVPN network.
- **show nve peers**: Displays information that determine if the VNI is configured for peer.

Show Command-BGP

```
#show ip bgp l2vpn evpn summary
BGP router identifier 19.0.0.1, local AS number 1
BGP table version is 2, main routing table version 2
1 network entries using 376 bytes of memory
1 path entries using 196 bytes of memory
1/1 BGP path/bestpath attribute entries using 272 bytes of memory
1 BGP extended community entries using 40 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 884 total bytes of memory
BGP activity 1/0 prefixes, 1/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
20::46          4            1    7852    7849        2    0    0 4d22h           0
19.0.101.1      4            1       0       0        1    0    0 never    Idle
19.0.101.2      4            1       0       0        1    0    0 never    Idle
19.0.101.3      4            1       0       0        1    0    0 never    Idle
19.0.101.4      4            1       0       0        1    0    0 never    Idle
19.0.101.5      4            2       0       0        1    0    0 never    Idle
19.0.101.6      4            1       0       0        1    0    0 never    Idle
19.0.101.7      4            1   80385    7853        2    0    0 4d22h           1
20.0.0.47       4            1    7857    7844        2    0    0 4d22h           0
FEC0::1001      4            1       0       0        1    0    0 never    Idle
```

Show Command-MLRIB

```
# show mlrib evpn mac
EVI   MAC Address  Owner Next-Hop  iVNI  eVNI
----------------------------------------------------
100   aaa.bbb.cc1   NVE   1.2.3.4   10000 1000

# show mlrib evpn mac detailed
EVI MAC Address Owner Next-Hop  iVNI  eVNI lVTEP     port
-------------------------------------------------------------
100 aaa.bbb.cc1 NVE   1.2.3.4   1000  1000 1.2.3.2   2000

# show mlrib evpn vtep local
BD    RMAC Address    VTEP-IP       VRF     VNI  BDI
----------------------------------------------------------
100   aaa.bbb.cc2    101.2.3.4      vrf1    10000    BDI100
```

Show NVE Peers

```
#sh nve peers vni 10135
Interface  VNI      Type Peer-IP          Router-RMAC    eVNI     state flags UP time
nve1       10135    L3CP 66.66.66.66      5c83.8f5f.5c97 10135      UP   A/M 00:08:53
```

Configuring EVPN: Basic Configuration

Perform the following tasks to configure EVPN:

1. Create a VRF.

```
vrf definition EVPN
 rd 100:1
 !
 address-family ipv4
  route-target export 100:1 stitching
  route-target import 100:1 stitching
 exit-address-family
```

2. Create a bridge domain and assign a VNI.

```
bridge-domain 1234
 member vni 101234
```

3. Create a BDI interface and assign it to the EVPN VRF.

```
interface BDI1234
 vrf forwarding EVPN
 ip address 10.20.30.40 255.255.255.0
 encapsulation dot1Q 1234
```

4. Create an NVE interface.

```
interface nve1
 no ip address
 source-interface Loopback1
 host-reachability protocol bgp
 member vni 101234 vrf EVPN

router bgp 100
 bgp router-id 10.10.10.10
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.10.10.111 remote-as 100
 neighbor 10.10.10.111 ebgp-multihop 255
 neighbor 10.10.10.111 update-source Loopback1
 neighbor 10.10.10.222 remote-as 100
 neighbor 10.10.10.222 ebgp-multihop 255
 neighbor 10.10.10.222 update-source Loopback1
 !
```

5. Configure a EVPN sessions to two spines.

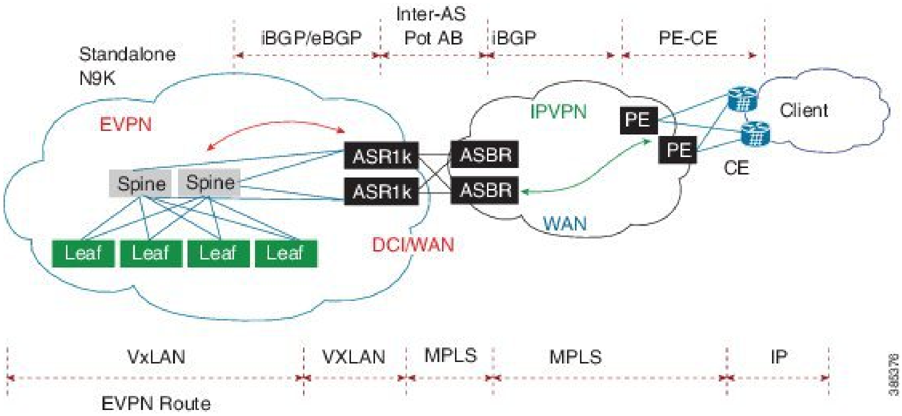
```
address-family l2vpn evpn
  neighbor 10.10.10.111 activate
  neighbor 10.10.10.111 send-community both
  neighbor 10.10.10.222 activate
  neighbor 10.10.10.222 send-community both
 exit-address-family
```

EVPN Interconnect With MPLS VPN as ASBR

In the scenario explained in the below figure shows, EVPN routes from the DC side get imported into VRFs at the ASR1k. These routes are in turn re-originated to the WAN side MPLS core network via VPN routes to

ASBR using a variation of MPLS-VPN Inter-AS option AB. There is only 1 BGP peering between the ASR1k and the ASBR, but the forwarding happens on multiple VRF sub-interfaces.

Figure 6: EVPN Interconnect With MPLS VPN as ASBR



Configuring Inter-AS Option AB

The following sections describe how to configure the Inter-AS Option AB feature on an ASBR for either an MPLS VPN or an MPLS VPN that supports CSC:

Note If Inter-AS Option AB is already deployed in your network and you want to do Option B style peering for some prefixes (that is, implement Inter-AS Option AB+), configure the **inter-as-hybrid global** command as described in the “Configuring the Routing Policy for VPNs that Need Inter-AS Connections” section.

Configuring the VRFs on the ASBR Interface for Each VPN Customer

Use the following steps to configure the VRFs on the ASBR interface for each VPN customer so that these VPNs have connectivity over the MPLS VPN--Inter-AS Option AB network.

Note The **mpls bgp forwarding** command is used only on the ASBR interface for VRFs that support CSC.

Use all of the steps in the following procedure to configure additional VRFs that need to be configured on the ASBR interface and the VRFs that need to be configured on the peer ASBR interface.

1. Enable privileged EXEC mode. Enter your password if prompted.

```
enable
Example:
Router> enable
```

2. Enter global configuration mode.

```
configure terminal
Example:
Router# configure terminal
```

3. Specify the interface to configure and enter the interface configuration mode.

- The *type* argument specifies the type of interface to be configured.
- The *number* argument specifies the port, connector, or interface card number.

```
interface type number
Example:
 Router(config)# interface Ethernet 5/0
```

4. Associate a VRF with the specified interface or subinterface.
   - The *vrf-name* argument is the name assigned to a VRF.

```
ip vrf forwarding vrf-name
Example:
 Router(config-if)# ip vrf forwarding vpn1
```

5. (Optional) Configures BGP to enable MPLS forwarding on connecting interfaces for VRFs that must support MPLS traffic.
   - This step applies to a CSC network only.

```
mpls bgp forwarding
Example:
 Router(config-if)# mpls bgp forwarding
```

6. (Optional) Exits to privileged EXEC mode.

```
end
Example:
 Router(config-if)# end
```

Configuring MP-BGP Session Between ASBR Peers

BGP propagates reachability information for VPN-IPv4 prefixes among PE routers by means of the BGP multiprotocol extensions (see RFC 2283, *Multiprotocol Extensions for BGP-4*), which define support for address families other than IPv4. Using the extensions ensures that the routes for a given VPN are learned only by other members of that VPN, enabling members of the VPN to communicate with each other.

Follow the steps in this section to configure the MP-BGP session on the ASBR.

Use all of the steps in the following procedure to configure the MP BGP session on the peer ASBR.

1. Enable privileged EXEC mode. Enter your password if prompted.

```
enable
Example:
Router> enable
```

2. Enter global configuration mode.

```
configure terminal
Example:
Router# configure terminal
```

3. Configures a BGP routing process and places the router in router configuration mode.
   - The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.

```
router bgp as-number
Example:
 Router(config)# router bgp 100
```

4. Adds an entry to the BGP or multiprotocol BGP neighbor table.
   - The *ip-address* argument specifies the IP address of the neighbor.
   - The *peer-group-name* argument specifies the name of a BGP peer group.
   - The *as-number* argument specifies the autonomous system to which the neighbor belongs.

```
neighbor {ip-address | peer-group-name} remote-as as-number
Example:
 Router(config-router)# neighbor 192.168.0.1
remote-as 200
```

5. Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard VPNv4 address prefixes.
   - The **unicast** keyword specifies IPv4 unicast address prefixes.

```
address-family vpnv4 [unicast]
Example:
 Router(config-router)# address-family vpnv4
```

6. Enables the exchange of information with a neighboring router.
   - The *ip-address* argument specifies the IP address of the neighbor.
   - The *peer-group-name* argument specifies the name of a BGP peer group.

```
neighbor {ip-address | peer-group-name} activate
Example:
 Router(config-router-af)# neighbor 192.168.0.1
activate
```

7. Configures eBGP peer router (ASBR) as an Inter-AS Option AB peer.
   - The *ip-address* argument specifies the IP address of the neighbor.
   - The *peer-group-name* argument specifies the name of a BGP peer group.
   - If any prefixes are imported into Option AB VRFs, then the imported paths are advertised to this peer.
   - If any prefixes are received from this peer and are imported into Option AB VRFs, then the imported paths are advertised to iBGP peers.

Note Advertised routes have RTs that are configured on the VRF. Advertised routes do not have their original RTs.

```
neighbor {ip-address | peer-group-name} inter-as-hybrid
Example:
 Router(config-router-af)# neighbor 192.168.0.1
inter-as-hybrid
```

8. Exits from address family configuration mode.

```
exit-address-family
Example:
 Router(config-router-af)# exit-address-family
```

9. (Optional) Exits to privileged EXEC mode.

```
end
Example:
 Router(config-af)# end
```

Configuring the Routing Policy for VPNs that Need Inter-AS Connections

Use the steps in this section to configure VRFs for the VPNs that need Inter-AS connections between ASBR peers, by configuring the appropriate routing policy and Option AB configuration.

Use all of the steps in the following procedure to configure additional VPNs that need Inter-AS Option AB connectivity on this ASBR and the peer ASBR.

1. Enable privileged EXEC mode. Enter your password if prompted.

```
enable
Example:
Router> enable
```

2. Enter global configuration mode.

```
configure terminal
Example:
Router# configure terminal
```

3. Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode.

   - The *vrf-name* argument is the name assigned to a VRF.

```
vrf definition vrf-name
Example:
 Router(config)# vrf definition vpn1
```

4. Creates routing and forwarding tables.

   - The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter an RD in either of these formats:
     - 16-bit autonomous system number: your 32-bit number, for example, 101:3
     - 32-bit IP address: your 16-bit number, for example, 192.168.122.15:1

```
rd route-distinguisher
Example:
 Router(config-vrf)# rd 100:1
```

5. Enters VRF address family configuration mode to specify an address family for a VRF.

   - The **ipv4** keyword specifies an IPv4 address family for a VRF.

```
address-family ipv4
Example:
Router(config-vrf)# address-family ipv4
```

6. Creates a route-target extended community for a VRF.

   - The **import** keyword imports routing information from the target VPN extended community.

   - The **export** keyword exports routing information to the target VPN extended community.

- The **both** keyword imports routing information from and exports routing information to the target VPN extended community.
- The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF list of import, export, or both (import and export) route-target extended communities.

```
route-target {import | export | both}
route-target-ext-community
Example:
 Router(config-vrf-af)# route-target import
100:1
```

7. For Inter-AS Option AB+, go to Step 10; otherwise, go to Step 8.

8. Specifies the VRF as an Option AB VRF, which has the following effects:

- Routes imported to this VRF can be advertised to Option AB peers and VPNv4 iBGP peers.
- When routes received from Option AB peers and are imported into the VRF, the next hop table ID of the route is set to the table ID of the VRF.
- If the **csc** keyword is not used, a per-VRF label is allocated for imported routes.
- When routes are received from Option AB peers and are imported next into the VRF, the learned out label can be installed only in forwarding when the **csc** keyword is used.

The **csc** keyword implies the following:

- A per-prefix label is allocated for imported routes.
- For routes received from Option AB peers that are imported into the VRF, the learned out label is installed in forwarding.

```
inter-as-hybrid [csc]
Example:
 Router(config-vrf-af)# inter-as-hybrid
```

9. (Optional) Specifies the next hop IP address to be set on paths that are imported into the VRF and that are received from an Option AB peer.

- The next hop context is also set to the VRF, which imports these paths.
- The **csc** keyword implies the following:
  - A per-prefix label is allocated for imported routes.
  - For routes received from Option AB peers that are imported into the VRF, the learned out label is installed in forwarding.

```
inter-as-hybrid next-hop global
Example:
 Router(config-vrf-af)# inter-as-hybrid next-hop
global
```

10. (For Option AB+) Enables Inter-AS Option AB+.

- Specifies that the next-hop address for BGP updates to be set on paths that are imported to the VRF and that are received from an Option AB+ peer are placed in the global routing table.

- The address used is the address of the interface that is at the remote end of the external BGP (eBGP) global shared link. The next-hop context is retained as global and not modified to that of the importing VRF.

```
inter-as-hybrid next-hop global
Example:
 Router(config-vrf-af)# inter-as-hybrid next-hop
global
```

11. (Optional) Exits to privileged EXEC mode.

```
end
Example:
 Router(config-vrf-af)# end
```

Example: EVPN Interconnect With MPLS VPN as ASBR

```
router bgp 100
 bgp router-id 10.10.10.10
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 9.9.8.8 remote-as 200
 Neighbor 9.9.8.8 ebgp-multihop 255

 neighbor 9.9.8.8 update-source Loopback0
 !
 address-family vpnv4
  import l2vpn evpn
  neighbor 9.9.8.8 activate
  neighbor 9.9.8.8 send-community extended
  neighbor 9.9.8.8 next-hop-self all
  Neighbor 9.9.8.8 inter-as-hybrid
```

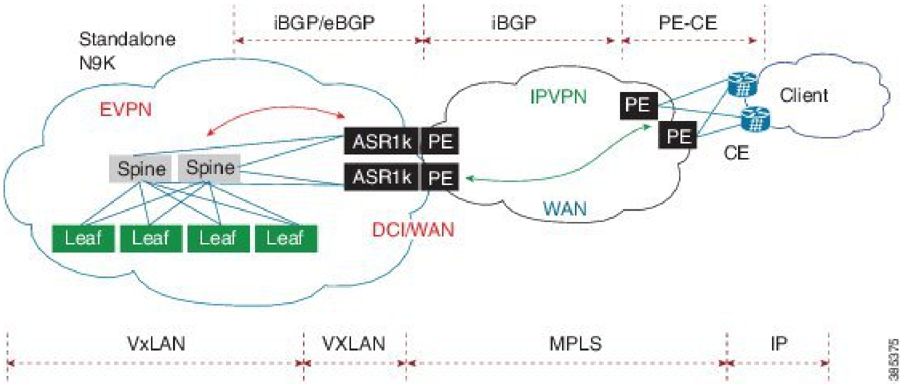
Configuring EVPN Interconnect With MPLS VPN as PE

ASR1000 supports direct prefix redistribution between BGP VPNv4 and BGP L2VPN EVPN address families. ASR1000 can act as gateway of Data Centre network and PE of MPLS VPN network both. It receives MPLS VPN prefixes from P/PE routers and these prefixes can be imported into BGP EVPN rib and then forwarded to DC's spine via BGP EVPN session. It can also import BGP EVPN prefixes sent by spine into BGP VPNv4 rib and send to P/PE in MPLS VPN network. During the prefixes redistribution, ASR1k set itself as the next-hop of the prefix before sending update to its neighbors.

In this release (16.4.1), ASR1000 only supports only bi-directional redistribution between EVPN and VPNv4. Redistribution between EVPN and VPNv6 is not supported.

In the scenario explained in the below figure shows, ASR1k acting as a PE in the MPLS-VPN network. Firstly, VRF is needed for the EVPN RT-5 routes to be imported, and then re-originate as VPN route into the MPLS-VPN side. VPN route that is learnt from the MPLS-VPN side will then first be imported into VRF, and the re-originated into EVPN as RT-5 routes.

Figure 7: EVPN Interconnect With MPLS VPN as PE



1. Define VRF and IPv4 address family.

```
vrf definition EVPN
 rd 100:1
 !
 address-family ipv4
  route-target import 100:1
  route-target import 100:1
  route-target export 100:1 stitching
  route-target import 100:1 stitching
 exit-address-family
!
```

2. Configure interface Loopback0.

```
interface Loopback0
MPLS VPN
 ip address 9.9.10.10 255.255.255.255
 ip router isis vpn
 Ip ospf 100 area 0
!
```

3. Configure interface GigabitEthernet.

```
interface GigabitEthernet0/0/0
facing MPLS VPN P/PE
 ip address 9.9.108.10 255.255.255.0
 ip router isis vpn
 negotiation auto
 mpls ip
cdp enable
!
Interface gi0/0/1.4
  Description facing to ACI spine
 Encapsulation dot1q 4
  Ip address 10.10.10.1 255.255.255.0
  Ip ospf 100 area 0
```

4. Create Interface NVE1.

```
interface nve1
 no ip address
 source-interface Loopback0
 host-reachability protocol bgp
 member vni 101234 vrf EVPN
!
```

5. Configure bridge domain.

```
Bridge-domain 100
  Member vni 101234
Interface bdi100
  Vrf forwarding EVPN
   Encapsulation dot1q 100
 Ip address 9.10.0.1 255.255.255.0

Router ospf 100
  Router-id 9.9.10.10
  Area 0.0.0.100 nssa

router isis vpn
 net 49.0001.1010.1010.1010.00
 is-type level-2-only
 metric-style wide
!
```

6. Define BGP and EVPN address-family.

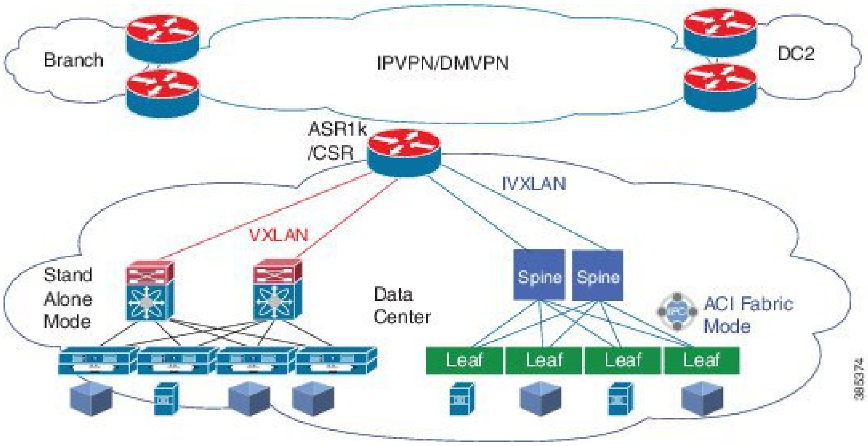
```
router bgp 200
 bgp router-id 10.10.10.10
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 9.9.8.8 remote-as 200
 neighbor 9.9.8.8 update-source Loopback0
 neighbor 10.10.10.111 remote-as 100
 neighbor 10.10.10.111 ebgp-multihop 255
 neighbor 10.10.10.111 update-source Loopback0
 neighbor 10.10.10.222 remote-as 100
 neighbor 10.10.10.222 ebgp-multihop 255
 neighbor 10.10.10.222 update-source Loopback0
 !
 address-family vpnv4
  import l2vpn evpn
  neighbor 9.9.8.8 activate
  neighbor 9.9.8.8 send-community extended
  neighbor 9.9.8.8 next-hop-self all
 exit-address-family
 !
 address-family l2vpn evpn
  import vpnv4 unicast
  neighbor 10.10.10.111 activate
  neighbor 10.10.10.111 send-community both
  neighbor 10.10.10.222 activate
  neighbor 10.10.10.222 send-community both
 exit-address-family
 !
```

7. Define VXLAN UDP port.

```
vxlan udp port 0xBEEF
```

Configuring DCI EVPN Peer to ACI Spine

Figure 8: DCI EVPN Peer to ACI Spine



1. Configure interface.

```
Interface gi0/0/1.4
  Description facing to ACI spine
 Encapsulation dot1q 4
  Ip address 10.10.10.1 255.255.255.0
  Ip ospf 100 area 0
```

2. Configure bridge domain.

```
Bridge-domain 100
  Member vni 101234
Interface bdi100
  Vrf forwarding EVPN
   Encapsulation dot1q 100
 Ip address 9.10.0.1 255.255.255.0

Router ospf 100
  Router-id 9.9.10.10
  Area 0.0.0.100 nssa
vxlan udp port 0xBEEF
```

Additional References for EVPN VxLAN L3

MIBs

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

Technical Assistance

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

Feature Information for EVPN VxLAN L3

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

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

Table 35: Feature Information for EVPN VxLAN L3

Feature Name	Releases	Feature Information
EVPN VxLAN L3	Cisco IOS XE Denali 16.3.1	The EVPN VxLAN L3 is a new feature.
VXLAN EVPN Fabric DCI - MPLS L3VPN	Cisco IOS XE Everest 16.4.1	The VXLAN EVPN Fabric DCI - MPLS L3VPN is a new feature.

CHAPTER 26

OpFlex Configuration

OpFlex Control Protocol or OpFlex is an open source protocol that supports policy exchange between network policy controller and smart devices that are capable of rendering abstract policies.

Restrictions for OpFlex Configuration

- Only two peers per domain are supported on opflex.
- Loopback or physical interface address can be used for opflex agent configuration on the device, but only physical interface ip address can be used for the ACI fabric.
- Avoid configuring VRFs on the device that are getting pushed through opflex.
- If multiple peers exist on the device, and the opflex agent receives the tenant update from multiple peers for the same tenant, the opflex agent picks the latest update from the opflex server.
- VLAN 4 is used only for peering between ASR1000 device and the ACI fabric.

Feature Information for OpFlex Configuration

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

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

Table 36: Feature Information for OpFlex Configuration

Feature Name	Releases	Feature Information
OpFlex Configuration	Cisco IOS XE Release 16.5	The OpFlex Control Protocol or OpFlex Configuration is an open source protocol that supports policy exchange between the network policy controller and smart devices that are capable of rendering abstract policies. The following commands were introduced or modified by this feature: **opflex agent**, **service vxlan-evpn**.

Opflex Agent

Conbody

OpFlex is used to communicate policy from a border leaf to a router configured as a datacenter interconnect. In this configuration, the ASR1000 device is considered to have a ACI fabric facing side that peers with a border leaf over OpFlex and a WAN facing side that is configured either manually or through a separate network management system or controller.

With this configuration, we can automate frequently changing per-tenant configurations through OpFlex. The ACI information model does not support a complete end-to-end WAN configuration. For example, parameters that may be exchanged between a border leaf and DCI device include a tenant ID, VRF ID, ASN ID, and DCI-IP.

Note You can configure a DCI device and gather WAN statistics over opflex using abstract policy.

Opflex Agent in Polaris

The role of the Opflex Agent (OFA) is to handle policy updates, and render configuration as required. To handle updates, the agent needs to establish a connection with an Opflex peer. Then, as the updates start from the peer(s), the agent operates in two phases: extracts data from the received policy and auto-generates the concrete configuration.

Note OFA is simultaneously used for multiple configuration models that control multiple functionalities. To facilitate this, OFA uses the notion of service, which handles a subset of incoming policies and renders concrete configuration appropriately.

For more information on the EVPN configuration, refer to the *EVPN VxLAN L3* section in the Carrier Ethernet Configuration Guide:

Peering and Opflex Control Protocol

Peering and core opflex protocol support is provided by a suite of third-party libraries: `libopflex, libuv, boost, and rapidjson`. The major roles are played by the `libopflex` library, which is responsible for the opflex protocol implementation, and `libuv`, which is responsible for the networking connections.

The opflex protocol uses TCP or IP for communication. Packets are sent and received through the data plane by leveraging the LFTS layer which allows BINOS processes to use TCP or IP sockets.. The peer ip-address and tcp-port are set through IOS configuration commands and used to provision the connection.

For each peer, once a connection is established, policy updates are pushed to the opflex agent from the Controller through the peer. At present time there is no way to send an indication to the peer or controller about the success of handling a policy update. That is, from the controller's view, all policy updates are successful. This is a limitation imposed by `libopflex`.

Policy data extraction

The policies are encoded in JSON format. It is the role of libopflex to parse JSON encoded policies and call an appropriate handler to process the data. This data pertained in the policy is service-specific, that is, it is used to configure a particular networking solution and thus requires service-specific handling.

Each service provides a dedicated BINOS side parser which is used by OFA to extract the policy data, used later for configuration rendering. The invocation of these parser is the responsibility of the Domain MGR.

Services

Different services are implemented as distinct sub-systems and can be packaged according to specific platform requirements. If a service is packaged, it is required to register with the Service MGR in OFA. During registration, the service provides several callbacks which are used by OFA to interact with the service.

The role of a service is to take the extracted policy data and apply the logic contained in it to produce the required concrete auto-generated configuration. The actual configuration rendering is performed in IOS. Similar to policy data extraction, rendering the concrete configuration requires service-specific implementation.

OFA also provides several event notifications to services. The handling of these events are within the service responsibility. The notified events are:

- Removal of a domain
- NVGEN request
- RP becoming active (following a switchover)

Policy Data Delivery to Services

As the data extraction and configuration rendering take place in two different components, with the former being on the BINOS side and the latter in IOS, the policy data needs to be delivered from the service-specific parser to the corresponding OFA service in IOS.

Therefore, while maintaining the generic nature of the OFA infrastructure, the policy data is transported as payload in a generic datagram. That is, the OFA infrastructure is not made aware of the service-specific data, which allows a simple addition of supported services.

Peering

For peering information OFA uses the notion of a domain. Domain is a logical collection of one or two peers which are synchronized. That is, two peers cannot send the same policy update.

Having two peers adds to fault resistance of the particular domain. Below is an example of an Opflex domain configuration.

```
Opflex agent
  Domain dom1
       Identity dci-[10.20.30.40]
       Peer p1 ip-address 156.2.21.10 tcp-port 8009 src-ip-address 148.2.15.1
       Peer p2 ip-address 156.2.21.10 tcp-port 8009 src-ip-address 148.2.15.1
```

Domain: At most two peers per domain.

Identity: The identity is a string that uniquely identifies a domain and needs to be matched with the Opflex peer definition of a domain identity, which is governed by the Opflex library in use.

The third party library (used by OFA), `libopflex`, uses the format `dci-[ip]`, where `ip` is the ip-address of the bgp ip loopback protocol.

Peers. The required information for each peer is the `ip-address` and `tcp-port` of the peer. The `src-ip-address` is the ip-address used by the OFA to send Opflex requests.

OFA Infra

The peering information and services configuration is synced to the standby through the common CONFIG_SYNC mechanism. The active peering connections are not established on the standby, but are re-created following a switchover.

There is no additional state that needs to be tracked.

OFA services

It is the responsibility of the service to support HA. To help facilitate this, OFA provides several workflows to support various HA scenarios.

Incremental sync following a policy update

1. The policy data is delivered to the service on the active
2. The policy data is handled on the active

3. The policy data is augmented by the service with auto-generated values and repackaged
4. OFA sends the augmented policy data to the standby
5. The augmented policy data is delivered to the service

Bulk Sync on Standby

- OFA requests the service for the current state in the form of a list of policy data items (which will recreate the state)
- OFA sends the list to the standby
- Each policy data item on the list is delivered to the service

Domain Removal CLI

- Domain is removed on the active (through CLI)
- Service is notified of a domain removal on the active
- Domain is removed on the standby (through CONFIG_SYNC)
- Service is notified of a domain removal on the standby
- Each service can choose the respective parts of the workflow

Active	Standby	Comments
Peering information	As on the active	Through CONFIG_SYNC
Peering connections	Disabled	Re-established on switchover
Individual services CLI configuration	As on the active	Through CONFIG_SYNC
Service state (e.g. updates history)	As supported by service	
Service generated configuration	As supported by service	

Configuring the Opflex Agent

OFA Configuration has two parts - the generic peering information and service-specific setup. The former creates logical policy update sources (2 peers per source), while the latter provides a platform for a service to have its particular configuration data.

Opflex Agent Services

Services are used to configure particular feature sets and the one service supported by OFA is the Service VXLAN-EVPN, or in short SVE.

Service VXLAN-EVPN (SVE)

Details about the actual service and the business logic it provides can be found in a document describing the VXLAN-EVPN solution. Here we focus on the service functionality from the Opflex perspective, i.e. service configuration and the rendering template.

In a nutshell, SVE receives policy updates which hold tenant information and render the corresponding configuration based on a template. The policy updates are encoded in the JSON format and convey several pieces of information:

- Policy space
- Routing domain
- Local VRF name
- A list of route targets (RTs)

Following updates, SVE builds an internal Tenant DB with the following properties for each tenant:

- Unique name (based on policy space and routing domain)
- Route targets (RTs)
- VRF name

Each tenant contributes its list of RTs to the VRF it references (name received in the policy update). Note that the same VRF can be referenced by multiple tenants, while each tenant references only a single VRF.

Each policy update carries information for a single tenant. The update can be of two types:

- **Tenant PUT** completely replaces current information in the Tenant DB with the new one.
- **Tenant DELETE** removes the tenant from the Tenant DB.

The rendered configuration is an up-to-date projection of the Tenant DB into VRF, BD, BGP, and NVE features. For each referenced VRF the rendering template in Fig. 3 is used

The rendered configuration is an up-to-date projection of the Tenant DB into VRF, BD, BGP, and NVE features. For each referenced VRF, the rendering template is used.

Services Configuration

SVE is configured as a subtree under opflex agent. It has two configurable items:

- nve-id: The interfaceis used for VNI generation
- bdi-ip: It is used for the configuration of the generated BDI
- Both the items are used in the rendering template

Manual versus Rendered Configuration

At times the rendered and manual configuration might have a conflict. The following set of rules outlines the current behavior of SVE in various scenarios where there is a potential collision between an auto-generated and manual config.

- SVE configuration may overwrite existing manual configuration for BDI attributes (ip address/forwarding vrf).
- Manual config is allowed to overwrite any SVE-generated configuration. Note that if done improperly service disruption is possible.
- Additional features, such as, NAT or QoS are manually configured on an SVE-generated BDI.
- SVE never deletes VRF/BD/BDI/NVE definitions (including those which are auto-generated following a tenant update)
- The only cleanup allowed by SVE is the RTs (normal or stitching) configured inside the VRF by SVE. That is, SVE will delete RTs during tenant updates. As a result we might have a rare scenario where a user configured RT is deleted by SVE due to tenant update if the tenant had a similar RT before the update.

```
Opflex agent
    Service vxlan-evpn
        Nve-id 1
        Bdi-ip 9.9.9.9 255.255.255.0
```

Policy Data Delivery to Services

As the data extraction and configuration rendering take place in two different components, with the former being on the BINOS side and the latter in IOS, the policy data needs to be delivered from the service-specific parser to the corresponding OFA service in IOS.

To achieve that, while maintaining the generic nature of the OFA infrastructure, the policy data is transported as payload in a generic datagram. That is, the OFA infrastructure is not made aware of the service-specific data, which allows a simple addition of supported services.

High Availability for Opflex

There are two main aspects for High Availability in SVE: tenant information and features (VRF/BD/BDI/NVE) configurational data. Features sync config data independently from SVE. Whereas SVE utilizes the OFA infra to sync Tenant DB information (see Section 0 on OFA High Availability)

In terms of data consistency: on the active, tenant information fully matches features' config data; on the standby, however, tenant information is not tied to features' config data as both are synced independently (for example a VRF might not yet be available on the standby when tenant information is synced).

After switchover each tenant is validated to have all the matching feature configuration. In case some pieces are missing (which might happen during the feature sync process), these are re-configured.

Handling of OFA events

As mentioned previously, each services is notified of a system-wide event. For the purposes of this user guide, it is of interest to discuss the consequences of domain removal and switchover in SVE.

Peering

When a domain is removed (unconfigured), SVE deletes the tenants which were last modified through that domain. For example, in the case of 4 tenants (as shown in the table below), if domain D1 is removed, this would lead to deletion of tenants T1, and T3.

Tenant deletion does not alter any of the existing configuration, except for route targets, as indicated in Section 0). Also, recall that the generated concrete configuration is a projection of the Tenant DB. Thus RTs will remain part of a VRF as long as there at least one tenant in the DB which still holds the RT in question.

Switchover

As mentioned earlier, the features are responsible for successful syncing of their respective configuration to the standby (VRF, BD, BGP, NVE).

On switchover, however, SVE attempts to re-apply the expected configuration. For each tenant, if successful, this results in no-op, otherwise, the tenant is deleted.

Configuring OpFlex Configuration

```
Device(config)# interface FortyGigabitEthernet1/1/1.4
 description "Connected to Spine"
 encapsulation dot1Q 4
 ip address 88.0.0.4 255.255.255.0
 ip ospf mtu-ignore
!
interface Loopback0
 ip address 1.1.1.1 255.255.255.255
!
router ospf 100
 nsf ietf
 area 1 nssa
 area 100 nssa
 network 1.1.1.1 0.0.0.0 area 0
 network 1.0.0.0 0.0.0.255 area 100
!
vrf definition cust2
rd 1:1054
!
address-family ipv4
 route-target export 100:2
 route-target import 100:2
exit-address-family
!
bridge-domain 10368
 member vni 15000001
!
bridge-domain 10367
 member vni 15000017
!
interface bdi 10368
 vrf forwarding cust1
 ip address 1.1.1.1 255.255.255.0
 no shutdown
!
interface bdi 10367
 vrf forwarding cust2
 ip address 1.1.1.1 255.255.255.0
 no shutdown
!
opflex agent
service vxlan-evpn
  nve-id 1
  bdi-ip 1.1.1.1 255.255.255.0
domain Fabric1
  identity dci-[31.1.1.1]
peer 1 ip-address 88.0.0.3 tcp-port 8009 src-ip-address 88.0.0.4
!
interface nve1
 no ip address
 source-interface Loopback0
 host-reachability protocol bgp
 vxlan udp port 48879
!
router bgp 101
 bgp router-id 31.1.1.1
 bgp log-neighbor-changes
 bgp listen limit 5000
 bgp graceful-restart
 timers bgp 120 360
 neighbor 102.102.102.102 remote-as 100
 neighbor 102.102.102.102 ebgp-multihop 255
 neighbor 102.102.102.102 update-source Loopback0
 neighbor 102.102.102.102 ha-mode graceful-restart
!
 address-family ipv4
  neighbor 102.102.102.102 activate
 exit-address-family
 !
 address-family l2vpn evpn
  import vpnv4 unicast re-originate
  neighbor 102.102.102.102 activate
  neighbor 102.102.102.102 send-community both
 exit-address-family
!
```

Example: Opflex Configuration

Example: OpFlex Configuration

OpFlex Configuration

```
Device# show opflex agent service vxlan-evpn all
Number of tenants: 2
Domain Name Tenant Name VRF
BD VNI
DC1 acivrf_cust1 cust1
10368 15000001
DC1 evpn2_cust2 cust2
10367 15000017
DeviceI# show opflex agent service vxlan-evpn tenant evpn2_cust2
Tenant: evpn2_cust2 (domain DC1)
VRF: cust2 (id 5, rd_nn 1054)
BDI: 10367 VNI: 15000017
Route Targets:
 RT:100:2 ipv4 import
 RT:100:2 ipv4 export
Device# show opflex agent service vxlan-evpn concrete-config all
Number of configs: 2
vrf definition cust1
 rd 1:1053
 address-family ipv4 unicast
 route-target import RT:100:1 stitching
 route-target import RT:100:1
 route-target export RT:100:1 stitching
 route-target export RT:100:1
 address-family ipv6 unicast
 route-target import RT:100:1 stitching
 route-target import RT:100:1
 route-target export RT:100:1 stitching
 route-target export RT:100:1
vrf definition cust2
 rd 1:1054
 address-family ipv4 unicast
 route-target export RT:100:2 stitching
 route-target export RT:100:2
 route-target import RT:100:2 stitching
 route-target import RT:100:2
 address-family ipv6 unicast
 route-target export RT:100:2 stitching
 route-target export RT:100:2
 route-target import RT:100:2 stitching
 route-target import RT:100:2
bridge-domain 10368
 member vni 15000001
interface bdi 10368
 vrf forwarding cust1
 ip address 100.1.1.1 255.255.255.0
 no shutdown
bridge-domain 10367
 member vni 15000017
interface bdi 10367
 vrf forwarding cust2
 ip address 100.1.1.1 255.255.255.0
no shutdown
router bgp 101
 address-family ipv4 vrf cust1
  advertise l2vpn evpn
 address-family ipv4 vrf cust2
  advertise l2vpn evpn

interface nve 1
 member vni 15000001 vrf cust1
 member vni 15000017 vrf cust2
Device# show ip route vrf cust2
Routing Table: cust2
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
 D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
 N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
 E1 - OSPF external type 1, E2 - OSPF external type 2
 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
 ia - IS-IS inter area, * - candidate default, U - per-user static
route
 o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
 a - application route
 + - replicated route, % - next hop override, p - overrides from PfR
Gateway of last resort is not set
 5.0.0.0/32 is subnetted, 1 subnets
S 5.1.2.2 is directly connected, Null0
 15.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
```

```
B 15.0.0.1/32 [20/0] via 30.1.2.2, 00:35:56
B 15.1.1.0/24 [20/0] via 30.1.2.2, 00:35:56
 30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C 30.1.2.0/24 is directly connected, TenGigabitEthernet2/1/0.2
L 30.1.2.1/32 is directly connected, TenGigabitEthernet2/1/0.2
 50.0.0.0/24 is subnetted, 51 subnets
B 50.1.2.0 [20/0] via 10.0.0.34, 3d07h, BDI10367
B 50.2.1.0 [20/0] via 10.0.0.34, 3d07h, BDI10367
60.0.0.0/24 is subnetted, 1 subnets
B 60.1.2.0 [20/0] via 30.1.2.2, 00:35:56
 64.0.0.0/24 is subnetted, 1 subnets
B 64.1.2.0 [20/0] via 30.1.2.2, 00:35:56
 65.0.0.0/24 is subnetted, 1 subnets
B 65.1.2.0 [20/0] via 30.1.2.2, 00:35:56
 100.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C 100.1.1.0/24 is directly connected, BDI10367
L 100.1.1.1/32 is directly connected, BDI10367
```

Verifying How to Display or Cleanup Tenants

Verifying OpFlex Configuration

The **show** or **clear** commands are used to display to clean-up tenants.

```
Device# show opflex agent service vxlan-evpn all
Prints brief tenant information for the tenants in the Tenant DB. Information includes:
    Domain name - the most recent source for tenant configuration
    Tenant name - the name of the tenant which is a combination of Opflex Policy Space and
 Routing Domain
    VRF - the tenant VRF
    BD/VNI - the bridge domain/VNI information associated with the tenant
show opflex agent service vxlan-evpn tenant tenant-name
    Prints detailed a specific tenant which includes route targets.
    Device# show opflex agent service vxlan-evpn concrete-config all
    Prints the concrete configuration associated with the tenant

Device# clear opflex agent service vxlan-evpn all
Deletes all tenants currently in the Tenant DB. Note that the tenants do not get automatically
 re-configured. To re-create the tenants, the user has to flap recreate the domain(s). Also,
 the deletion only takes place only on the RP where it is executed (e.g. deleting tenants
on the active does not delete the tenants on the standby).
Debug commands
debug opflex agent service vxlan-evpn [all|debug|error|info]
Enable different levels of debugging for SVE.
```

Additional References for OpFlex Configuration

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Standards and RFCs

Standard/RFC	Title
Standard	*Title*

MIBs

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

Technical Assistance

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