# Configuring Ethernet OAM

This module describes the configuration of Ethernet Operations, Administration, and Maintenance (OAM) .

**Feature History for Configuring Ethernet OAM**

| Release | Modification |
|---------|--------------|
|         |              |

# Information About Configuring Ethernet OAM

To configure Ethernet OAM, you should understand the following concepts:

## Ethernet Link OAM

**Table 1: Feature History Table**

Ethernet as a Metro Area Network (MAN) or a Wide Area Network (WAN) technology benefits greatly from the implementation of Operations, Administration and Maintenance (OAM) features. Ethernet link OAM features allow Service Providers to monitor the quality of the connections on a MAN or WAN. Service providers can monitor specific events, . Ethernet link OAM operates on a single, physical link and it can be configured to monitor either side or both sides of that link.

Ethernet link OAM can be configured in the following ways:

- A Link OAM profile can be configured, and this profile can be used to set the parameters for multiple interfaces.

- Link OAM can be configured directly on an interface.

  When an interface is also using a link OAM profile, specific parameters that are set in the profile can be overridden by configuring a different value directly on the interface.

An Ethernet Link OAM profile simplifies the process of configuring EOAM features on multiple interfaces. An Ethernet OAM profile, and all of its features, can be referenced by other interfaces, allowing other interfaces to inherit the features of that Ethernet OAM profile.

Individual Ethernet link OAM features can be configured on individual interfaces without being part of a profile. In these cases, the individually configured features always override the features in the profile.

The preferred method of configuring custom EOAM settings is to create an EOAM profile in Ethernet configuration mode and then attach it to an individual interface or to multiple interfaces.

When an EOAM packet is received on any one of the AC interfaces on which EOAM is not configured, the AC interface multicasts the received EOAM packets to other AC interfaces that are part of EVPN-BD to reach the peer. When an EOAM is enabled on the bundle member in the peer, it punts the packet to the CPU in the peer. Also, the EOAM flaps the bundle member as the local or remote Key of the received EOAM does not match.

These standard Ethernet Link OAM features are supported on the router:

## Neighbor Discovery

Neighbor discovery enables each end of a link to learn the OAM capabilities of the other end and establish an OAM peer relationship. Each end also can require that the peer have certain capabilities before it will establish a session. You can configure certain actions to be taken if there is a capabilities conflict or if a discovery process times out, using the **action capabilities-conflict** or **action discovery-timeout** commands.

## EFD

Ethernet Fault Detection (EFD) is a mechanism that allows Ethernet OAM protocols, such as CFM, to control the `line protocol` state of an interface.

Unlike many other interface types, Ethernet interfaces do not have a line protocol, whose state is independent from that of the interface. For Ethernet interfaces, this role is handled by the physical-layer Ethernet protocol itself, and therefore if the interface is physically up, then it is available and traffic can flow.

EFD changes this to allow CFM to act as the line protocol for Ethernet interfaces. This allows CFM to control the interface state so that if a CFM defect (such as AIS or loss of continuity) is detected with an expected peer MEP, the interface can be shut down. This not only stops traffic flow, but also triggers actions in any higher-level protocols to route around the problem. For example, in the case of Layer 2 interfaces, the MAC table would be cleared and MSTP would reconverge. For Layer 3 interfaces, the ARP cache would be cleared and potentially the IGP would reconverge.

> **Note** EFD can only be used for down MEPs. When EFD is used to shut down the interface, the CFM frames continue to flow. This allows CFM to detect when the problem has been resolved, and thus bring the interface backup automatically.

This figure shows CFM detection of an error on one of its sessions EFD signaling an error to the corresponding MAC layer for the interface. This triggers the MAC to go to a down state, which further triggers all higher level protocols (Layer 2 pseudowires, IP protocols, and so on) to go down and also trigger a reconvergence where possible. As soon as CFM detects there is no longer any error, it can signal to EFD and all protocols will once again go active.

*Figure 1: CFM Error Detection and EFD Trigger*
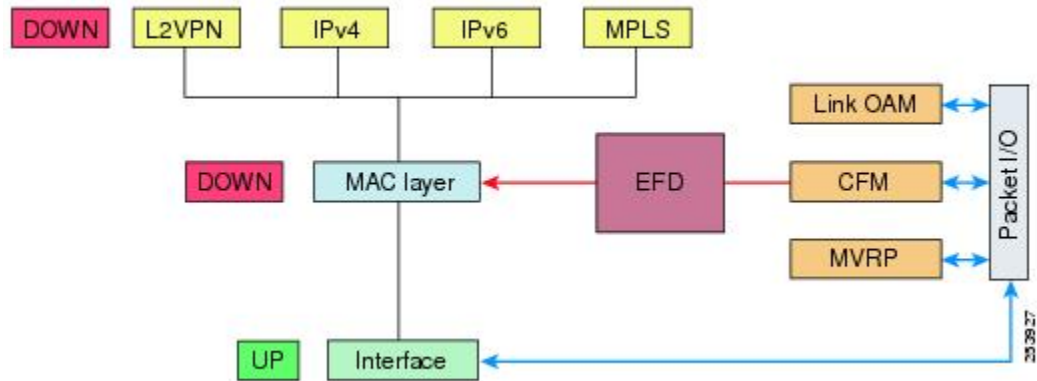


## MIB Retrieval

MIB retrieval enables an OAM peer on one side of an interface to get the MIB variables from the remote side of the link. The MIB variables that are retrieved from the remote OAM peer are READ ONLY.

## SNMP Traps

SNMP traps can be enabled or disabled on an Ethernet OAM interface.

## Unidirectional Link Detection Protocol

Unidirectional Link Detection (UDLD) is a single-hop physical link protocol for monitoring an ethernet link, including both point-to-point and shared media links. This is a Cisco-proprietary protocol to detect link problems, which are not detected at the physical link layer. This protocol is specifically targeted at possible wiring errors, when using unbundled fiber links, where there can be a mismatch between the transmitting and receiving connections of a port.

### Limitations

- UDLD is not tunneled through L2VPN like other slow protocols.

- UDLD must not be enabled on a Switched Port Analyzer (SPAN) source or a destination port.

- UDLD must not be enabled on a port that acts as a source or destination port for SPAN.

# Ethernet CFM

Ethernet Connectivity Fault Management (CFM) is a service-level OAM protocol that provides tools for monitoring and troubleshooting end-to-end Ethernet services per VLAN. This includes proactive connectivity monitoring, fault verification, and fault isolation. CFM uses standard Ethernet frames and can be run on any physical media that is capable of transporting Ethernet service frames. Unlike most other Ethernet protocols which are restricted to a single physical link, CFM frames can transmit across the entire end-to-end Ethernet network.

✎

**Note**    Enable a maximum of 32 VLAN ranges per NPU. Else, when you reload the device, all CFM sessions over the 802.1Q VLAN interface might go down. Also, the corresponding bundle interface might go down. If more than 32 VLAN ranges exist on an NPU, remove the additional VLAN ranges and reload the device to address the issue.

CFM is defined in two standards:

- IEEE 802.1ag—Defines the core features of the CFM protocol.

- ITU-T Y.1731—Redefines, but maintains compatibility with the features of IEEE 802.1ag, and defines some additional features.

Ethernet CFM supports these functions of ITU-T Y.1731:

- ETH-CC, ETH-RDI, ETH-LB, ETH-LT—These are equivalent to the corresponding features defined in IEEE 802.1ag.

✎

**Note**    The Linktrace responder procedures defined in IEEE 802.1ag are used rather than the procedures defined in Y.1731; however, these are interoperable.

- ETH-AIS—The reception of ETH-LCK messages is also supported.

To understand how the CFM maintenance model works, you need to understand these concepts and features:

## Maintenance Domains

A maintenance domain describes 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 interfaces internal to it and at its boundary, as shown in this figure.

**Figure 2: CFM Maintenance Domain**
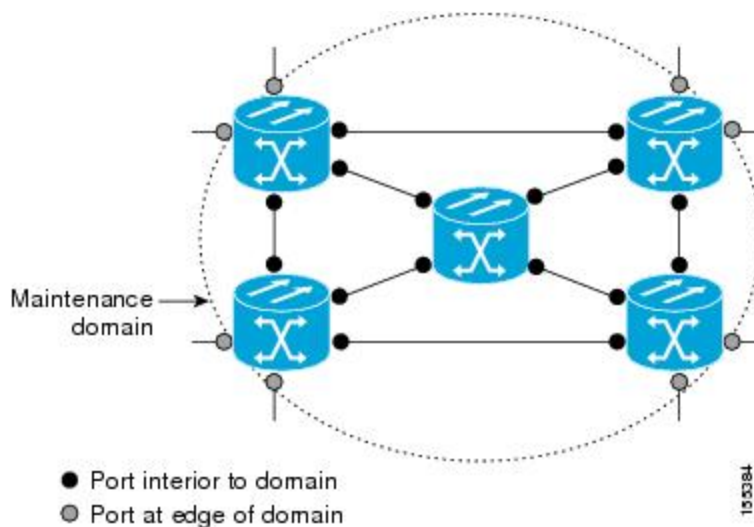
A maintenance domain is defined by the bridge ports that are provisioned within it. Domains are assigned maintenance levels, in the range of 0 to 7, by the administrator. The level of the domain is useful in defining the hierarchical relationships of multiple domains.

CFM maintenance domains allow different organizations to use CFM in the same network, but independently. For example, consider a service provider who offers a service to a customer, and to provide that service, they use two other operators in segments of the network. In this environment, CFM can be used in the following ways:

- The customer can use CFM between their CE devices, to verify and manage connectivity across the whole network.

- The service provider can use CFM between their PE devices, to verify and manage the services they are providing.

- Each operator can use CFM within their operator network, to verify and manage connectivity within their network.

Each organization uses a different CFM maintenance domain.

This figure shows an example of the different levels of maintenance domains in a network.

**Note** In CFM diagrams, the conventions are that triangles represent MEPs, pointing in the direction that the MEP sends CFM frames, and circles represent MIPs.

**Figure 3: Different CFM Maintenance Domains Across a Network**
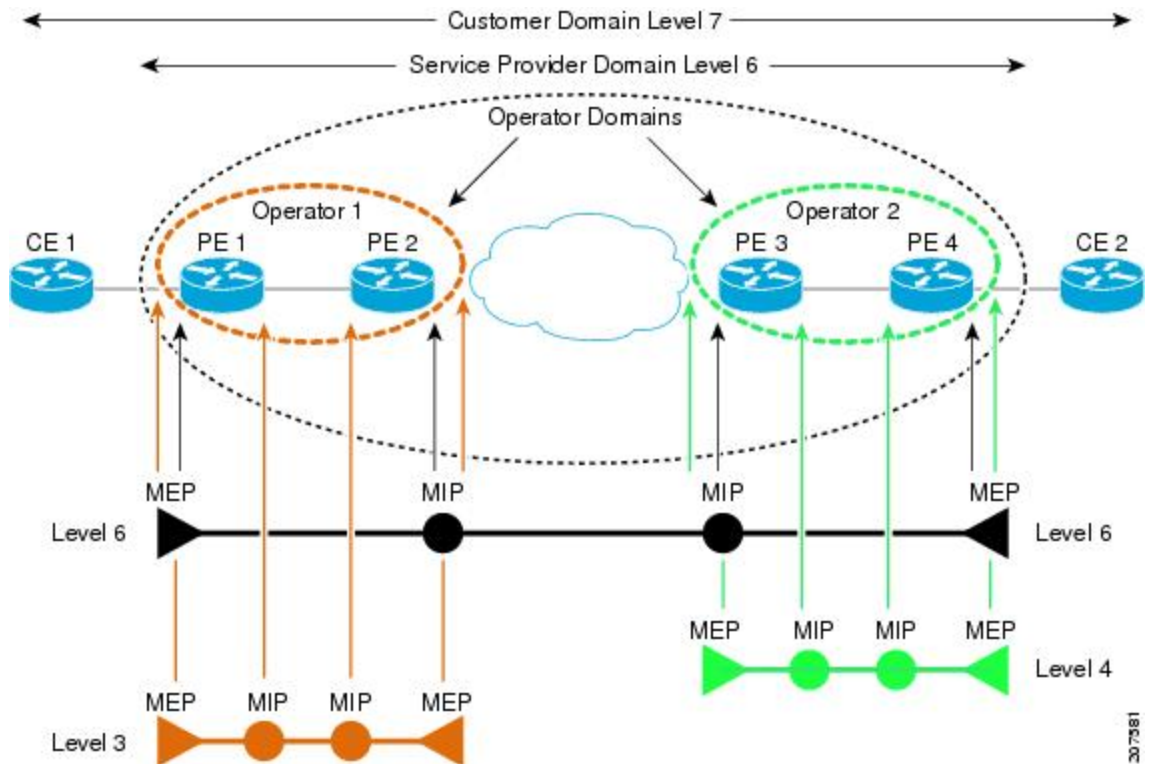
To ensure that the CFM frames for each domain do not interfere with each other, each domain is assigned a maintenance level, between 0 and 7. Where domains are nested, as in this example, the encompassing domain must have a higher level than the domain it encloses. In this case, the domain levels must be negotiated between the organizations involved. The maintenance level is carried in all CFM frames that relate to that domain.

CFM maintenance domains may touch or nest, but cannot intersect. This figure illustrates the supported structure for touching and nested domains, and the unsupported intersection of domains.



Scenario A:
Touching Domains OK

Scenario B:
Nested Domains OK

Scenario C:
Intersecting Domains
Not Allowed
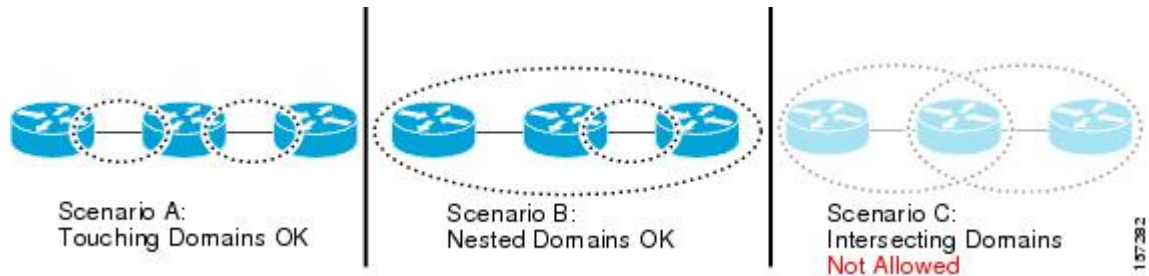
## Services

A CFM service allows an organization to partition its CFM maintenance domain, according to the connectivity within the network. For example, if the network is divided into a number of virtual LANs (VLANs), a CFM service is created for each of these. CFM can then operate independently in each service. It is important that the CFM services match the network topology, so that CFM frames relating to one service cannot be received in a different service. For example, a service provider may use a separate CFM service for each of their customers, to verify and manage connectivity between that customer's end points.

A CFM service is always associated with the maintenance domain that it operates within, and therefore with that domain's maintenance level. All CFM frames relating to the service carry the maintenance level of the corresponding domain.

**Note**   CFM Services are referred to as *Maintenance Associations* in IEEE 802.1ag and as *Maintenance Entity Groups* in ITU-T Y.1731.

## Maintenance Points

A CFM Maintenance Point (MP) is an instance of a particular CFM service on a specific interface. CFM only operates on an interface if there is a CFM maintenance point on the interface; otherwise, CFM frames are forwarded transparently through the interface.

A maintenance point is always associated with a particular CFM service, and therefore with a particular maintenance domain at a particular level. Maintenance points generally only process CFM frames at the same level as their associated maintenance domain. Frames at a higher maintenance level are always forwarded transparently, while frames at a lower maintenance level are normally dropped. This helps enforce the maintenance domain hierarchy, and ensures that CFM frames for a particular domain cannot leak out beyond the boundary of the domain.

There are two types of MP:

- Maintenance End Points (MEPs)—Created at the edge of the domain. Maintenance end points (MEPs) are members of a particular service within a domain and are responsible for sourcing and sinking CFM frames. They periodically transmit continuity check messages and receive similar messages from other

MEPs within their domain. They also transmit traceroute and loopback messages at the request of the administrator. MEPs are responsible for confining CFM messages within the domain.

- Maintenance Intermediate Points (MIPs)—Created in the middle of the domain. Unlike MEPS, MIPs do allow CFM frames at their own level to be forwarded.

## MIP Creation

Unlike MEPs, MIPs are not explicitly configured on each interface. MIPs are created automatically according to the algorithm specified in the CFM 802.1ag standard. The algorithm, in brief, operates as follows for each interface:

- The bridge-domain or cross-connect for the interface is found, and all services associated with that bridge-domain or cross-connect are considered for MIP auto-creation.

- The level of the highest-level MEP on the interface is found. From among the services considered above, the service in the domain with the lowest level that is higher than the highest MEP level is selected. If there are no MEPs on the interface, the service in the domain with the lowest level is selected.

- The MIP auto-creation configuration (**mip auto-create** command) for the selected service is examined to determine whether a MIP should be created.

**Note** Configuring a MIP auto-creation policy for a service does not guarantee that a MIP will automatically be created for that service. The policy is only considered if that service is selected by the algorithm first.

## MEP and CFM Processing Overview

The boundary of a domain is an interface, rather than a bridge or host. Therefore, MEPs can be sub-divided into two categories:

- Down MEPs—Send CFM frames from the interface where they are configured, and process CFM frames received on that interface. Down MEPs transmit AIS messages upward (toward the cross-connect).

- Up MEPs—Send frames into the bridge relay function, as if they had been received on the interface where the MEP is configured. They process CFM frames that have been received on other interfaces, and have been switched through the bridge relay function as if they are going to be sent out of the interface where the MEP is configured. Up MEPs transmit AIS messages downward (toward the wire). However, AIS packets are only sent when there is a MIP configured on the same interface as the MEP and at the level of the MIP.

**Note**
- The terms *Down MEP* and *Up MEP* are defined in the IEEE 802.1ag and ITU-T Y.1731 standards, and refer to the direction that CFM frames are sent from the MEP. The terms should not be confused with the operational status of the MEP.

- The router only supports the "Down MEP level < Up MEP level" configuration.

This figure illustrates the monitored areas for Down and Up MEPs.

Figure 4: Monitored Areas for Down and Up MEPs
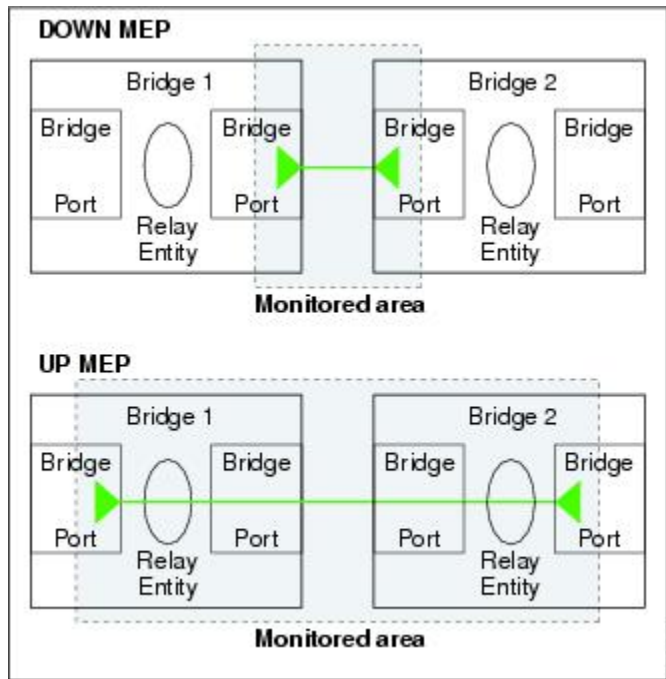


This figure shows maintenance points at different levels. Because domains are allowed to nest but not intersect (see ), a MEP at a low level always corresponds with a MEP or MIP at a higher level. In addition, only a single MIP is allowed on any interface—this is generally created in the lowest domain that exists at the interface and that does not have a MEP.
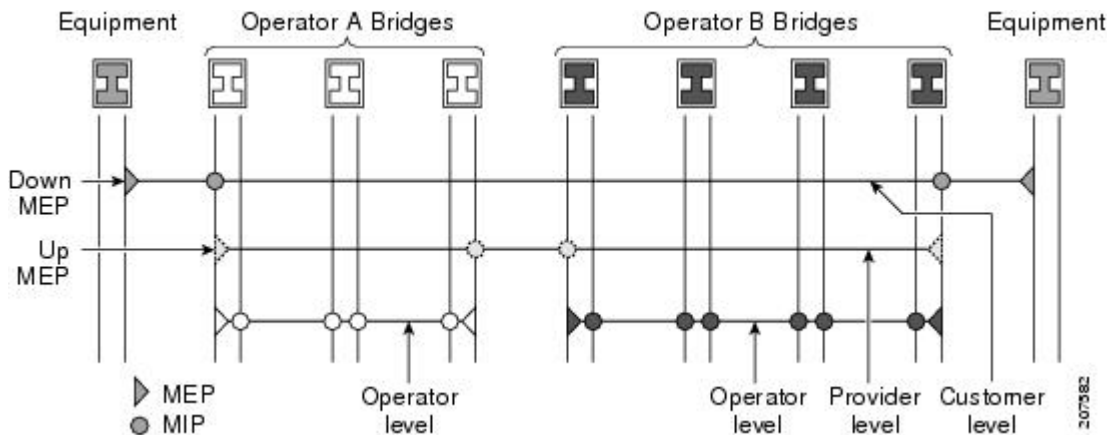


MIPs and Up MEPs can only exist on switched (Layer 2) interfaces, because they send and receive frames from the bridge relay function. Down MEPs can be created on switched (Layer 2) interfaces.

MEPs continue to operate normally if the interface they are created on is blocked by the Spanning Tree Protocol (STP); that is, CFM frames at the level of the MEP continue to be sent and received, according to the direction of the MEP. MEPs never allow CFM frames at the level of the MEP to be forwarded, so the STP block is maintained.

MIPs also continue to receive CFM frames at their level if the interface is STP blocked, and can respond to any received frames. However, MIPs do not allow CFM frames at the level of the MIP to be forwarded if the interface is blocked.

**Note** A separate set of CFM maintenance levels is created every time a VLAN tag is pushed onto the frame. Therefore, if CFM frames are received on an interface which pushes an additional tag, so as to "tunnel" the frames over part of the network, the CFM frames will not be processed by any MPs within the tunnel, even if they are at the same level. For example, if a CFM MP is created on an interface with an encapsulation that matches a single VLAN tag, any CFM frames that are received at the interface that have two VLAN tags will be forwarded transparently, regardless of the CFM level.

# CFM Protocol Messages

The CFM protocol consists of a number of different message types, with different purposes. All CFM messages use the CFM EtherType, and carry the CFM maintenance level for the domain to which they apply.

This section describes the following CFM messages:

## Continuity Check (IEEE 802.1ag and ITU-T Y.1731)

Continuity Check Messages (CCMs) are "heartbeat" messages exchanged periodically between all the MEPs in a service. Each MEP sends out multicast CCMs, and receives CCMs from all the other MEPs in the service—these are referred to as *peer MEPs*. This allows each MEP to discover its peer MEPs, and to verify that there is connectivity between them.

MIPs also receive CCMs. MIPs use the information to build a MAC learning database that is used when responding to Linktrace. For more information about Linktrace, see the Linktrace (IEEE 802.1ag and ITU-T Y.1731).

*Figure 5: Continuity Check Message Flow*
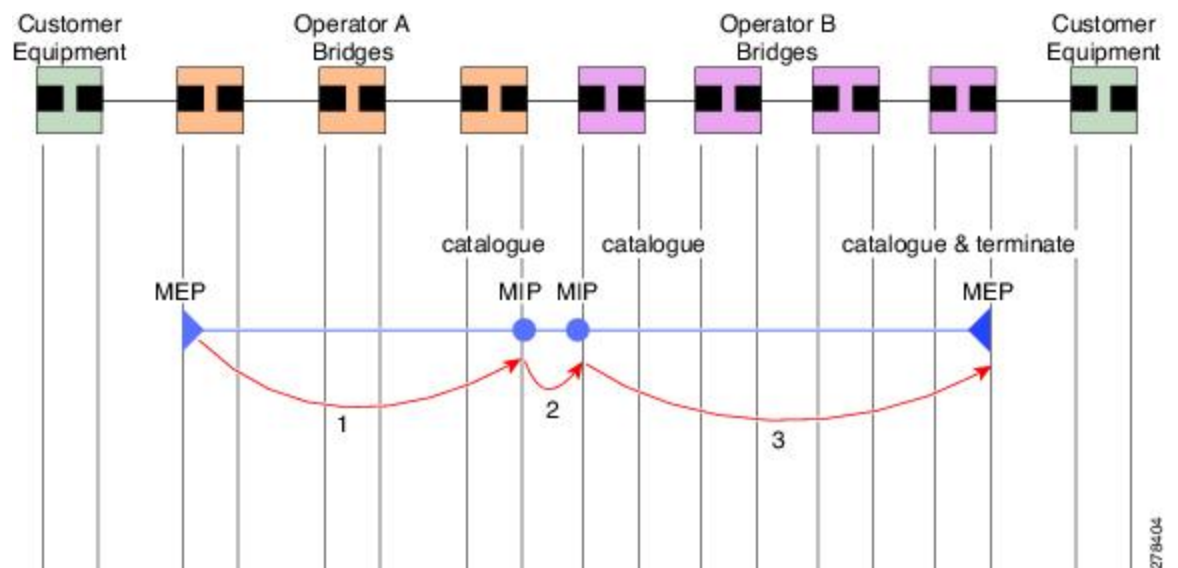


All the MEPs in a service must transmit CCMs at the same interval. IEEE 802.1ag defines 7 possible intervals that can be used:

- 3.3ms

- 10ms

- 100ms

- 1s

- 10s

- 1 minute

A MEP detects a loss of connectivity with one of its peer MEPs when some number of CCMs have been missed. This occurs when sufficient time has passed during which a certain number of CCMs were expected, given the CCM interval. This number is called the *loss threshold*, and is usually set to 3.

CFM is supported only on interfaces which have Layer 2 transport feature enabled.

## Maintenance Association Identifier (MAID)

CCM messages carry a variety of information that allows different defects to be detected in the service. This information includes:

- A configured identifier for the domain of the transmitting MEP. This is referred to as the Maintenance Domain Identifier (MDID).

- A configured identifier for the service of the transmitting MEP. This is referred to as the Short MA Name (SMAN). Together, the MDID and the SMAN make up the Maintenance Association Identifier (MAID). The MAID must be configured identically on every MEP in the service.

- These are restrictions on the type of MAID that are supported for sessions with time interval of less than 1 minute. The MAID supports two types of formats on offloaded MEPs:

  - No Domain Name Format

    - MD Name Format = 1-NoDomainName

    - Short MA Name Format = 3 - 2 bytes integer value

    - Short MA NAme Length = 2 - fixed length

    - Short MA Name = 2 bytes of integer

  - 1731 Maid Format

    - MD Name Format = 1-NoDomainName

    - MA Name Format(MEGID Format) = 32

    - MEGID Length = 13 - fixed length

    - MEGID(ICCCode) = 6 Bytes

    - MEGID(UMC) = 7 Bytes

    - ITU Carrier Code (ICC) - Number of different configurable ICC code - 15 (for each NPU)

    - Unique MEG ID Code (UMC) - 4

Maintenance Association Identifier (MAID) comprises of the Maintenance Domain Identifier (MDID) and Short MA Name (SMAN).

MDID **only** supports **null** value and SMAN supports ITU Carrier Code (ICC) or a numerical. No other values are supported.

An example for configuring domain ID null is: **ethernet cfm domain SMB level 3 id null**

An example for configuring SMAN is: **ethernet cfm domain SMB level 3 id null service 901234AB xconnect group 99999 p2p 99999 id number 1**

The following table summarizes the supported values and parameters for MDID and SMAN. This table only details the MAID restriction on the hardware offload feature. There is no MAID restriction for software offload or non-offloaded MEPs.

| Format | MDID | SMAN | Support | Comment |
|---|---|---|---|---|
| | No | 2 byte integer | Yes | Up to 2000 entries |
| | No | 13 bytes ICCCode (6 bytes) and UMC (7 bytes) | Yes | Up to 15 unique ICC<br>Up to 4K UMC values |
| 48 bytes string based | 1-48 bytes of MDID and SMAN | | No | Most commonly used |

- A configured numeric identifier for the MEP (the MEP ID). Each MEP in the service must be configured with a different MEP ID.

- Dynamic Remote MEPs are not supported for MEPs with less than 1min interval. You must configure MEP CrossCheck for all such MEPS.

- Sequence numbering is not supported for MEPs with less than 1 minute interval.

- In a Remote Defect Indication (RDI), each MEP includes this in the CCMs it is sending, if it has detected a defect relating to the CCMs it is receiving. This notifies all the MEPs in the service that a defect has been detected somewhere in the service.

- The interval at which CCMs are being transmitted.

- CCM Tx/Rx statistics counters are not supported for MEPs with less than1 minute intervals.

- Sender TLV and Cisco Proprietary TLVs are not supported for MEPs with less than 1min intervals.

- The status of the interface where the MEP is operating—for example, whether the interface is up, down, STP blocked, and so on.

**Note** The status of the interface (up/down) should not be confused with the direction of any MEPs on the interface (Up MEPs/Down MEPs).

These defects can be detected from received CCMs:

- Interval mismatch—The CCM interval in the received CCM does not match the interval that the MEP is sending CCMs.

- Level mismatch—A MEP has received a CCM carrying a lower maintenance level than the MEPs own level.

- Loop—A CCM is received with the source MAC address equal to the MAC address of the interface where the MEP is operating.

- Configuration error—A CCM is received with the same MEP ID as the MEP ID configured for the receiving MEP.

- Cross-connect—A CCM is received with an MAID that does not match the locally configured MAID. This generally indicates a VLAN misconfiguration within the network, such that CCMs from one service are leaking into a different service.

- Peer interface down—A CCM is received that indicates the interface on the peer is down.

- Remote defect indication—A CCM is received carrying a remote defect indication.

**Note**   This defect does not cause the MEP to include a remote defect indication in the CCMs that it is sending.

Out-of-sequence CCMs can also be detected by monitoring the sequence number in the received CCMs from each peer MEP. However, this is not considered a CCM defect.

## Loopback (IEEE 802.1ag and ITU-T Y.1731)

Loopback Messages (LBM) and Loopback Replies (LBR) are used to verify connectivity between a local MEP and a particular remote MP. At the request of the administrator, a local MEP sends unicast LBMs to the remote MP. On receiving each LBM, the target maintenance point sends an LBR back to the originating MEP. Loopback indicates whether the destination is reachable or not—it does not allow hop-by-hop discovery of the path. It is similar in concept to an ICMP Echo (ping). Since loopback messages are destined for unicast addresses, they are forwarded like normal data traffic, while observing the maintenance levels. At each device that the loopback reaches, if the outgoing interface is known (in the bridge's forwarding database), then the frame is sent out on that interface. If the outgoing interface is not known, then the message is flooded on all interfaces.

This figure shows an example of CFM loopback message flow between a MEP and MIP.

*Figure 6: Loopback Messages*
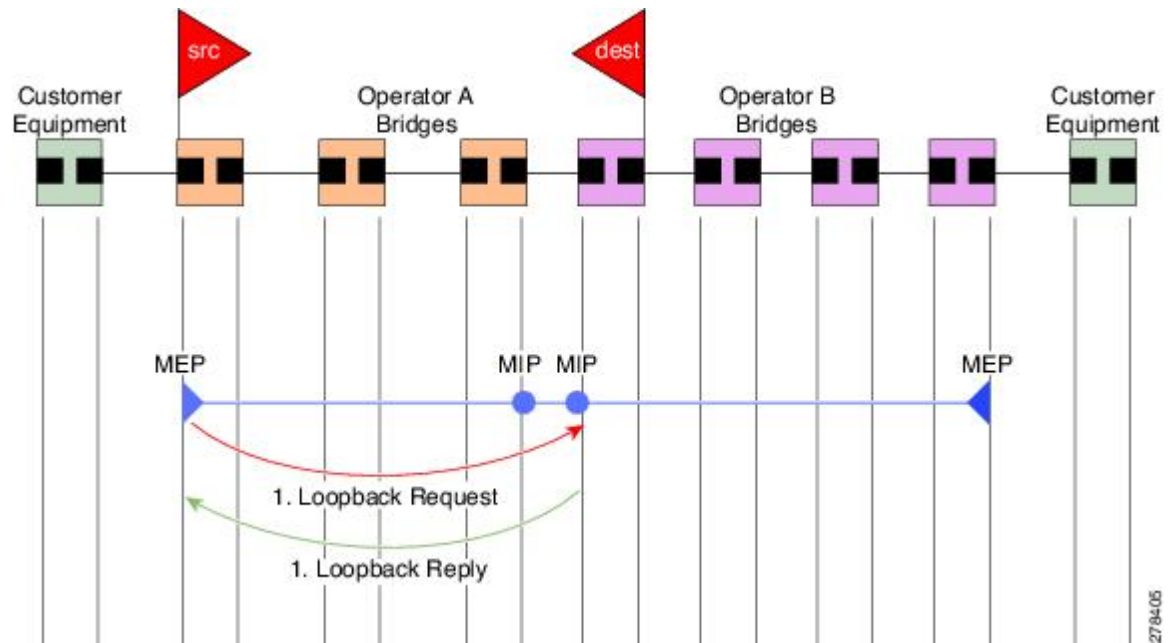


Loopback messages can be padded with user-specified data. This allows data corruption to be detected in the network. They also carry a sequence number which allows for out-of-order frames to be detected.

## Linktrace (IEEE 802.1ag and ITU-T Y.1731)

Linktrace Messages (LTM) and Linktrace Replies (LTR) are used to track the path (hop-by-hop) to a unicast destination MAC address. At the request of the operator, a local MEP sends an LTM. Each hop where there is a maintenance point sends an LTR back to the originating MEP. This allows the administrator to discover connectivity data about the path. It is similar in concept to IP traceroute, although the mechanism is different. In IP traceroute, successive probes are sent, whereas CFM Linktrace uses a single LTM which is forwarded by each MP in the path. LTMs are multicast, and carry the unicast target MAC address as data within the frame. They are intercepted at each hop where there is a maintenance point, and either retransmitted or dropped to discover the unicast path to the target MAC address.

This figure shows an example of CFM linktrace message flow between MEPs and MIPs.

*Figure 7: Linktrace Message Flow*
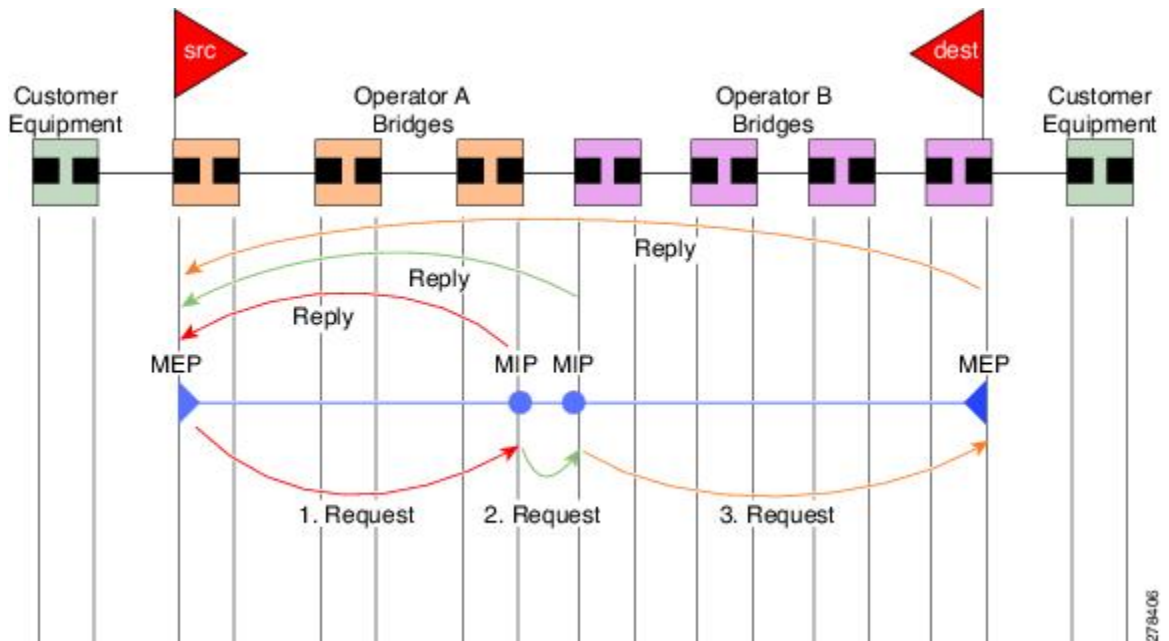


The linktrace mechanism is designed to provide useful information even after a network failure. This allows it to be used to locate failures, for example after a loss of continuity is detected. To achieve this, each MP maintains a CCM Learning Database. This maps the source MAC address for each received CCM to the interface through which the CCM was received. It is similar to a typical bridge MAC learning database, except that it is based only on CCMs and it times out much more slowly—on the order of days rather than minutes.

**Note**   In IEEE 802.1ag, the CCM Learning Database is referred to as the MIP CCM Database. However, it applies to both MIPs and MEPs.

In IEEE 802.1ag, when an MP receives an LTM message, it determines whether to send a reply using the following steps:

1. The target MAC address in the LTM is looked up in the bridge MAC learning table. If the MAC address is known, and therefore the egress interface is known, then an LTR is sent.

2. If the MAC address is not found in the bridge MAC learning table, then it is looked up in the CCM learning database. If it is found, then an LTR is sent.

3. If the MAC address is not found, then no LTR is sent (and the LTM is not forwarded).

If the target MAC has never been seen previously in the network, the linktrace operation will not produce any results.

![note icon]

**Note**     IEEE 802.1ag and ITU-T Y.1731 define slightly different linktrace mechanisms. In particular, the use of the CCM learning database and the algorithm described above for responding to LTM messages are specific to IEEE 802.1ag. IEEE 802.1ag also specifies additional information that can be included in LTRs. Regardless of the differences, the two mechanisms are interoperable.

## Configurable Logging

CFM supports logging of various conditions to syslog. Logging can be enabled independently for each service, and when the following conditions occur:

- New peer MEPs are detected, or loss of continuity with a peer MEP occurs.

- Changes to the CCM defect conditions are detected.

- Cross-check "missing" or "unexpected" conditions are detected.

- AIS condition detected (AIS messages received) or cleared (AIS messages no longer received).

- EFD used to shut down an interface, or bring it back up.

## Flexible VLAN Tagging for CFM

The Flexible VLAN Tagging for CFM feature ensures that CFM packets are sent with the right VLAN tags so that they are appropriately handled as a CFM packet by the remote device. When packets are received by an edge router, they are treated as either CFM packets or data packets, depending on the number of tags in the header. The system differentiates between CFM packets and data packets based on the number of tags in the packet, and forwards the packets to the appropriate paths based on the number of tags in the packet.

CFM frames are normally sent with the same VLAN tags as the corresponding customer data traffic on the interface, as defined by the configured encapsulation and tag rewrite operations. Likewise, received frames are treated as CFM frames if they have the correct number of tags as defined by the configured encapsulation and tag rewrite configuration, and are treated as data frames (that is, they are forwarded transparently) if they have more than this number of tags.

In most cases, this behavior is as desired, since the CFM frames are then treated in exactly the same way as the data traffic flowing through the same service. However, in a scenario where multiple customer VLANs are multiplexed over a single multipoint provider service (for example, N:1 bundling), a different behavior might be desirable.

This figure shows an example of a network with multiple VLANS using CFM.

*Figure 8: Service Provider Network With Multiple VLANs and CFM*
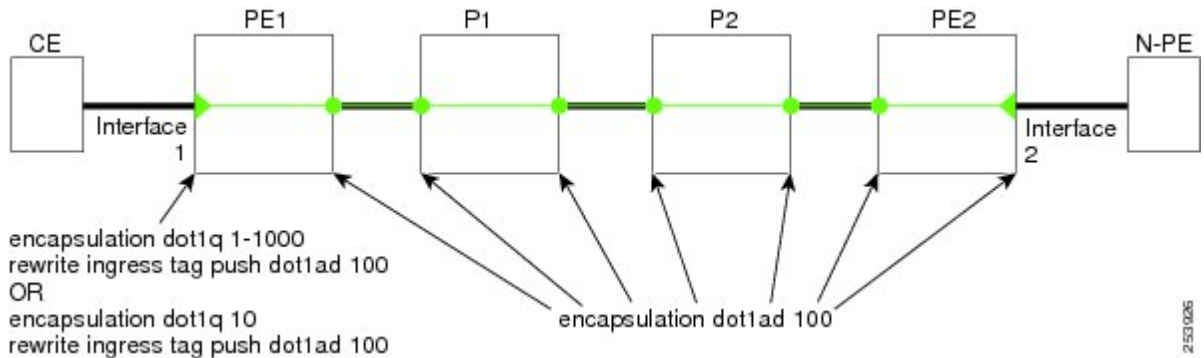


This figure shows a provider's access network, where the S-VLAN tag is used as the service delimiter. PE1 faces the customer, and PE2 is at the edge of the access network facing the core. N:1 bundling is used, so the interface encapsulation matches a range of C-VLAN tags. This could potentially be the full range, resulting in all:1 bundling. There is also a use case where only a single C-VLAN is matched, but the S-VLAN is nevertheless used as the service delimiter—this is more in keeping with the IEEE model, but limits the provider to 4094 services.

CFM is used in this network with a MEP at each end of the access network, and MIPs on the boxes within the network (if it is native Ethernet). In the normal case, CFM frames are sent by the up MEP on PE1 with two VLAN tags, matching the customer data traffic. This means that at the core interfaces and at the MEP on PE2, the CFM frames are forwarded as if they were customer data traffic, since these interfaces match only on the S-VLAN tag. So, the CFM frames sent by the MEP on PE1 are not seen by any of the other MPs.

Flexible VLAN tagging changes the encapsulation for CFM frames that are sent and received at Up MEPs. Flexible VLAN tagging allows the frames to be sent from the MEP on PE1 with just the S-VLAN tag that represents the provider service. If this is done, the core interfaces will treat the frames as CFM frames and they will be seen by the MIPs and by the MEP on PE2. Likewise, the MEP on PE1 should handle received frames with only one tag, as this is what it will receive from the MEP on PE2.

To ensure that CFM packets from Up MEPs are routed to the appropriate paths successfully, tags may be set to a specific number in a domain service, using the **tags** command. Currently, tags can only be set to one (1).

# How to Configure Ethernet OAM

This section provides these configuration procedures:

## Configuring Ethernet Link OAM

Custom EOAM settings can be configured and shared on multiple interfaces by creating an EOAM profile in Ethernet configuration mode and then attaching the profile to individual interfaces. The profile configuration does not take effect until the profile is attached to an interface. After an EOAM profile is attached to an interface, individual EOAM features can be configured separately on the interface to override the profile settings when desired.

This section describes how to configure an EOAM profile and attach it to an interface in these procedures:

## Configuring an Ethernet OAM Profile

Perform these steps to configure an Ethernet OAM profile.

**Procedure**

|  | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure terminal | Enters global configuration mode. |
| **Step 2** | **ethernet oam profile** *profile-name*<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# ethernet oam profile Profile_1 | Creates a new Ethernet Operations, Administration and Maintenance (OAM) profile and enters Ethernet OAM configuration mode. |
| **Step 3** | **link-monitor**<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-eoam)# link-monitor | Enters the Ethernet OAM link monitor configuration mode. |
| **Step 4** | **symbol-period window** *window*<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-eoam-lm)# symbol-period window 60000 | (Optional) Configures the window size (in milliseconds) for an Ethernet OAM symbol-period error event. The IEEE 802.3 standard defines the window size as a number of symbols rather than a time duration. These two formats can be converted either way by using a knowledge of the interface speed and encoding.<br><br>The range is 1000 to 60000.<br><br>The default value is 1000. |
| **Step 5** | **symbol-period threshold low** *threshold* **high** *threshold* **symbol-period threshold** { **ppm** [ **low** *threshold* ] [ **high** *threshold* ] | **symbols** [ **low** *threshold* [ **thousand** | **million** | **billion** ]] [ **high** *threshold* [ **thousand** | **million** | **billion** ]]}<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-eoam-lm)# symbol-period threshold ppm low 1 high 1000000 | (Optional) Configures the thresholds (in symbols) that trigger an Ethernet OAM symbol-period error event. The high threshold is optional and is configurable only in conjunction with the low threshold.<br><br>The range is 1 to 1000000.<br><br>The default low threshold is 1. |
| **Step 6** | **frame window** *window*<br>**Example:** | (Optional) Configures the frame window size (in milliseconds) of an OAM frame error event. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-eoam-lm)# frame window 6000` | The range is from 1000 to 60000. The default value is 1000. |
| Step 7 | **frame threshold low** *threshold* **high** *threshold* <br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)# frame threshold low 10000000 high 60000000` | (Optional) Configures the thresholds (in symbols) that triggers an Ethernet OAM frame error event. The high threshold is optional and is configurable only in conjunction with the low threshold. <br><br>The range is from 0 to 60000000. <br><br>The default low threshold is 1. |
| Step 8 | **frame-period window** *window* <br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)# frame-period window 60000`<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)# frame-period window milliseconds 60000` | (Optional) Configures the window size (in milliseconds) for an Ethernet OAM frame-period error event. The IEEE 802.3 standard defines the window size as number of frames rather than a time duration. These two formats can be converted either way by using a knowledge of the interface speed. Note that the conversion assumes that all frames are of the minimum size. <br><br>The range is from 1000 to 60000. <br><br>The default value is 1000. <br><br>**Note**  The only accepted values are multiples of the line cardinterface module-specific polling interval, that is, 1000 milliseconds for most line cardsinterface modules. |
| Step 9 | **frame-period threshold  low***threshold* **high** *threshold* **frame-period threshold** { **ppm** [ **low** *threshold* ] [ **high** *threshold* ] | **frames** [ **low** *threshold* [ **thousand** | **million** | **billion** ]] [ **high** *threshold* [ **thousand** | **million** | **billion** ]]} <br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)#    frame-period threshold ppm low 100  high 1000000` | (Optional) Configures the thresholds (in errors per million frames ) that trigger an Ethernet OAM frame-period error event. The frame period window is defined in the IEEE specification as a number of received frames, in our implementation it is x milliseconds. The high threshold is optional and is configurable only in conjunction with the low threshold. <br><br>The range is from 1 to 1000000. <br><br>The default low threshold is 1. <br><br>To obtain the number of frames, the configured time interval is converted to a window size in frames using the interface speed. For example, for a 1Gbps interface, the IEEE defines minimum frame size as 512 bits. So, we get a maximum of approximately 1.5 million frames per second. If the window size is configured |

| | Command or Action | Purpose |
|---|---|---|
| | | to be 8 seconds (8000ms) then this would give us a Window of 12 million frames in the specification's definition of Errored Frame Window. |
| | | The thresholds for frame-period are measured in errors per million frames. Hence, if you configure a window of 8000ms (that is a window of 12 million frames) and a high threshold of 100, then the threshold would be crossed if there are 1200 errored frames in that period (that is, 100 per million for 12 million). |
| **Step 10** | **frame-seconds window** *window*<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)# frame-seconds window 900000` | (Optional) Configures the window size (in milliseconds) for the OAM frame-seconds error event.<br><br>The range is 10000 to 900000.<br><br>The default value is 60000.<br><br>**Note**    The only accepted values are multiples of the line cardinterface module-specific polling interval, that is, 1000 milliseconds for most line cardsinterface modules. |
| **Step 11** | **frame-seconds threshold low** *threshold* **high** *threshold*<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)# frame-seconds threshold low 3 high 900` | (Optional) Configures the thresholds (in seconds) that trigger a frame-seconds error event. The high threshold value can be configured only in conjunction with the low threshold value.<br><br>The range is 1 to 900<br><br>The default value is 1. |
| **Step 12** | **exit**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam-lm)# exit` | Exits back to Ethernet OAM mode. |
| **Step 13** | **mib-retrieval**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam)# mib-retrieval` | Enables MIB retrieval in an Ethernet OAM profile or on an Ethernet OAM interface. |

| | | Command or Action | Purpose |
|---|---|---|---|
| Step 14 | | **connection timeout** <*timeout*> **Example:** `RP/0/RP0/CPU0:router(config-eoam)# connection timeout 30` | Configures the connection timeout period for an Ethernet OAM session. as a multiple of the hello interval. The range is 2 to 30. The default value is 5. |
| Step 15 | | **hello-interval** {**100ms**|**1s**} **Example:** `RP/0/RP0/CPU0:router(config-eoam)# hello-interval 100ms` | Configures the time interval between hello packets for an Ethernet OAM session. The default is 1 second (**1s**). |
| Step 16 | | **mode** {**active**|**passive**} **Example:** `RP/0/RP0/CPU0:router(config-eoam)# mode passive` | Configures the Ethernet OAM mode. The default is active. |
| Step 17 | | **require-remote mode** {**active**|**passive**} **Example:** `RP/0/RP0/CPU0:router(config-eoam)# require-remote mode active` | Requires that active mode or passive mode is configured on the remote end before the OAM session becomes active. |
| Step 18 | | **require-remote mib-retrieval** **Example:** `RP/0/RP0/CPU0:router(config-eoam)# require-remote mib-retrieval` | Requires that MIB-retrieval is configured on the remote end before the OAM session becomes active. |
| Step 19 | | **action capabilities-conflict** {**disable** | **efd** | **error-disable-interface**} **Example:** `RP/0/RP0/CPU0:router(config-eoam)# action capabilities-conflict efd` | Specifies the action that is taken on an interface when a capabilities-conflict event occurs. The default action is to create a syslog entry. **Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| Step 20 | | **action critical-event** {**disable** | **error-disable-interface**} **Example:** `RP/0/RP0/CPU0:router(config-eoam)#` | Specifies the action that is taken on an interface when a critical-event notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry. |

| | | Command or Action | Purpose |
|---|---|---|---|
| | | `action critical-event`<br>`error-disable-interface` | **Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 21** | | **action discovery-timeout** {**disable** \| **efd** \| **error-disable-interface**}<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam)#`<br>`action discovery-timeout efd` | Specifies the action that is taken on an interface when a connection timeout occurs. The default action is to create a syslog entry.<br><br>**Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 22** | | **action dying-gasp** {**disable** \| **error-disable-interface**}<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam)#`<br>`action dying-gasp`<br>`error-disable-interface` | Specifies the action that is taken on an interface when a dying-gasp notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.<br><br>**Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 23** | | **action high-threshold** {**error-disable-interface** \| **log**}<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam)#` | Specifies the action that is taken on an interface when a high threshold is exceeded. The default is to take no action when a high threshold is exceeded. |

| | Command or Action | Purpose |
|---|---|---|
| | `action high-threshold error-disable-interface` | **Note** • If you change the default, the **disable** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and take no action at the interface when the event occurs. |
| Step 24 | **action session-down** {**disable** \| **efd** \| **error-disable-interface**}<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam)# action session-down efd` | Specifies the action that is taken on an interface when an Ethernet OAM session goes down.<br><br>**Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| Step 25 | **action session-up disable**<br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-eoam)# action session-up disable` | Specifies that no action is taken on an interface when an Ethernet OAM session is established. The default action is to create a syslog entry.<br><br>**Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| Step 26 | **action uni-directional link-fault** {**disable** \| **efd** \| **error-disable-interface**} | Specifies the action that is taken on an interface when a link-fault notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.<br><br>**Note** • If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |

| | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 27** | **action wiring-conflict** {**disable** \| **efd** \| **log**}<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-eoam)#<br>action session-down efd | Specifies the action that is taken on an interface when a wiring-conflict event occurs. The default is to put the interface into error-disable state.<br><br>**Note**  • If you change the default, the **error-disable-interface** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and put the interface into error-disable state when the event occurs. |
| **Step 28** | **uni-directional link-fault detection**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-eoam)#<br>uni-directional link-fault detection | Enables detection of a local, unidirectional link fault and sends notification of that fault to an Ethernet OAM peer. |
| **Step 29** | **commit**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if)# commit | Saves the configuration changes to the running configuration file and remains within the configuration session. |
| **Step 30** | **end**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if)# end | Ends the configuration session and exits to the EXEC mode. |

## Attaching an Ethernet OAM Profile to an Interface

Perform these steps to attach an Ethernet OAM profile to an interface:

**Procedure**

| | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure terminal | Enters global configuration mode. |

| | | Command or Action | Purpose |
|---|---|---|---|
| Step 2 | | **interface** [ \| **HundredGigE**\| **TenGigE**] *interface-path-id* <br><br>**Example:** <br><br>`RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/0` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. |
| Step 3 | | **ethernet oam** <br><br>**Example:** <br><br>`RP/0/RP0/CPU0:router(config-if)# ethernet oam` | Enables Ethernet OAM and enters interface Ethernet OAM configuration mode. |
| Step 4 | | **profile** *profile-name* <br><br>**Example:** <br><br>`RP/0/RP0/CPU0:router(config-if-eoam)# profile Profile_1` | Attaches the specified Ethernet OAM profile (*profile-name*), and all of its configuration, to the interface. |
| Step 5 | | **commit** <br><br>**Example:** <br><br>`RP/0/RP0/CPU0:router(config-if)# commit` | Saves the configuration changes to the running configuration file and remains within the configuration session. |
| Step 6 | | **end** <br><br>**Example:** <br><br>`RP/0/RP0/CPU0:router(config-if)# end` | Ends the configuration session and exits to the EXEC mode. |

## Configuring Ethernet OAM at an Interface and Overriding the Profile Configuration

Using an EOAM profile is an efficient way of configuring multiple interfaces with a common EOAM configuration. However, if you want to use a profile but also change the behavior of certain functions for a particular interface, then you can override the profile configuration. To override certain profile settings that are applied to an interface, you can configure that command in interface Ethernet OAM configuration mode to change the behavior for that interface.

In some cases, only certain keyword options are available in interface Ethernet OAM configuration due to the default settings for the command. For example, without any configuration of the **action** commands, several forms of the command have a default behavior of creating a syslog entry when a profile is created and applied to an interface. Therefore, the **log** keyword is not available in Ethernet OAM configuration for these commands in the profile because it is the default behavior. However, the **log** keyword is available in Interface Ethernet OAM configuration if the default is changed in the profile configuration so you can retain the action of creating a syslog entry for a particular interface.

To see all of the default Ethernet OAM configuration settings, see the Verifying the Ethernet OAM Configuration.

To configure Ethernet OAM settings at an interface and override the profile configuration, perform these steps:

**Procedure**

|  | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure terminal | Enters global configuration mode. |
| **Step 2** | **interface** [HundredGigE | TenGigE] *interface-path-id*<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/0 | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.<br><br>**Note** • The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1. |
| **Step 3** | **ethernet oam**<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if)# ethernet oam | Enables Ethernet OAM and enters interface Ethernet OAM configuration mode. |
| **Step 4** | *interface-Ethernet-OAM-command*<br>**Example:**<br>RP/0/RP0/CPU0:router(config-if-eoam)# action capabilities-conflict error-disable-interface | Configures a setting for an Ethernet OAM configuration command and overrides the setting for the profile configuration, where *interface-Ethernet-OAM-command* is one of the supported commands on the platform in interface Ethernet OAM configuration mode. |
| **Step 5** | **commit**<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if)# commit | Saves the configuration changes to the running configuration file and remains within the configuration session. |
| **Step 6** | **end**<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if)# end | Ends the configuration session and exits to the EXEC mode. |

## Verifying the Ethernet OAM Configuration

Use the **show ethernet oam configuration** command to display the values for the Ethernet OAM configuration for a particular interface, or for all interfaces. The following example shows the default values for Ethernet OAM settings:

```
RP/0/RP0/CPU0:router# show ethernet oam configuration
Thu Aug  5 22:07:06.870 DST
GigabitEthernet0/0/0/0:
  Hello interval:                                        1s
  Mib retrieval enabled:                                  N
  Uni-directional link-fault detection enabled:           N
  Configured mode:                                   Active
  Connection timeout:                                     5
  Symbol period window:                                   0
  Symbol period low threshold:                            1
  Symbol period high threshold:                        None
  Frame window:                                        1000
  Frame low threshold:                                    1
  Frame high threshold:                                None
  Frame period window:                                 1000
  Frame period low threshold:                             1
  Frame period high threshold:                         None
  Frame seconds window:                               60000
  Frame seconds low threshold:                            1
  Frame seconds high threshold:                        None
  High threshold action:                               None
  Link fault action:                                    Log
  Dying gasp action:                                    Log
  Critical event action:                                Log
  Discovery timeout action:                             Log
  Capabilities conflict action:                         Log
  Wiring conflict action:                     Error-Disable
  Session up action:                                    Log
  Session down action:                                  Log
  Require remote mode:                               Ignore
  Require remote MIB retrieval:                           N
```

# Configuring Ethernet CFM

> **Note**  CFM is not supported for the following:

- L3 Interfaces and Sub-Interfaces

- Bundle Member Ports

- EVPN-FXC

- Bridge Domain

- VPLS

## Configuring a CFM Maintenance Domain

To configure a CFM maintenance domain, perform the following steps:

**Procedure**

|  | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# ethernet cfm | Enters Ethernet Connectivity Fault Management (CFM) configuration mode. |
| **Step 3** | **traceroute cache hold-time** *minutes* **size** *entries*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm)# traceroute cache hold-time 1 size 3000 | (Optional) Sets the maximum limit of traceroute cache entries or the maximum time limit to hold the traceroute cache entries. The default is 100 minutes and 100 entries. |
| **Step 4** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1 | Creates and names a container for all domain configurations and enters CFM domain configuration mode.<br><br>The level must be specified.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| **Step 5** | **end** or **commit**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn)# commit | Saves configuration changes.<br><br>• When you use the **end** command, the system prompts you to commit changes:<br><br>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br><br>• Entering **cancel** leaves the router in the current configuration session without |

| Command or Action | Purpose |
|---|---|
| | exiting or committing the configuration changes. |
| | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Services for a CFM Maintenance Domain

You can configure up to 2000 CFM services for a maintenance domain. To configure services for a CFM maintenance domain, perform the following steps:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# ethernet cfm | Enters Ethernet CFM configuration mode. |
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1 | Creates and names a container for all domain configurations at a specified maintenance level, and enters CFM domain configuration mode.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| **Step 4** | **service** *service-name* {**down-meps** | **xconnect group** *xconnect-group-name* **m2mp** | **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [ [**number** *number*]<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn)# service ABC xconnect group X1 p2p ADB | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain where MIPs and up MEPs will be created.<br><br>The **id** sets the short MA name. |
| **Step 5** | **end** or **commit**<br><br>**Example:** | Saves configuration changes. |

| Command or Action | Purpose |
|---|---|
| `RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# commit` | • When you use the **end** command, the system prompts you to commit changes:<br><br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br><br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Enabling and Configuring Continuity Check for a CFM Service

To configure Continuity Check for a CFM service, complete the following steps:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config)# ethernet cfm` | Enters Ethernet Connectivity Fault Management (CFM) configuration mode. |
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:** | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| Step 4 | **service** *service-name* {**down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [ [**number** *number*] **Example:** `RP/0/RP0/CPU0:router(config-cfm-dmn)# service ABC xconnect group X1 p2p ADB` | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created. The **id** sets the short MA name. |
| Step 5 | **continuity-check interval** *time* [**loss-threshold** *threshold*] **Example:** `RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# continuity-check interval 100m loss-threshold 10` | (Optional) Enables Continuity Check and specifies the time interval at which CCMs are transmitted or to set the threshold limit for when a MEP is declared down. |
| Step 6 | **continuity-check archive hold-time** *minutes* **Example:** `RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# continuity-check archive hold-time 100` | (Optional) Configures how long information about peer MEPs is stored after they have timed out. |
| Step 7 | **continuity-check loss auto-traceroute** **Example:** `RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# continuity-check loss auto-traceroute` | (Optional) Configures automatic triggering of a traceroute when a MEP is declared down. |
| Step 8 | **end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# commit` | Saves configuration changes.<br><br>• When you use the **end** command, the system prompts you to commit changes:<br><br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode |

| Command or Action | Purpose |
|---|---|
| | without committing the configuration changes. |
| | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |
| | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Automatic MIP Creation for a CFM Service

For more information about the algorithm for creating MIPs, see the **MIP Creation** section.

To configure automatic MIP creation for a CFM service, complete the following steps:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router# ethernet cfm` | Enters the Ethernet Connectivity Fault Management (CFM) configuration mode. |
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified. The only supported option is **id [null]** for less than 1min interval MEPS.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| **Step 4** | **service** *service-name* {**down-meps** \| **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based***icc-string umc-string*] \| [**number** *number*] | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or |

| | **Command or Action** | **Purpose** |
|---|---|---|
| | **Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn)# service ABC xconnect group X1 p2p ADB | associate the service with a bridge domain where MIPs and up MEPs will be created.<br><br>The **id** sets the short MA name. |
| **Step 5** | **mip auto-create** {**all** \| **lower-mep-only**} {**ccm-learning**}<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# mip auto-create all ccm-learning | (Optional) Enables the automatic creation of MIPs in a bridge domain. **ccm-learning** option enables CCM learning for MIPs created in this service. This must be used only in services with a relatively long CCM interval of at least 100 ms. CCM learning at MIPs is disabled by default. |
| **Step 6** | **end** or **commit**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# commit | Saves configuration changes.<br><br>• When you use the **end** command, the system prompts you to commit changes:<br><br>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br><br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Cross-Check on a MEP for a CFM Service

To configure cross-check on a MEP for a CFM service and specify the expected set of MEPs, complete the following steps:

**Procedure**

|  | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# ethernet cfm | Enters the Ethernet Connectivity Fault Management (CFM) configuration mode. |
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1 | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| **Step 4** | **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1 | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created.<br><br>The **id** sets the short MA name. |
| **Step 5** | **mep crosscheck**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-xcheck)# mep crosscheck mep-id 10 | Enters CFM MEP crosscheck configuration mode. |
| **Step 6** | **mep-id** *mep-id-number* [**mac-address** *mac-address*]<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-xcheck)# mep-id 10 | Enables cross-check on a MEP.<br><br>**Note** • Repeat this command for every MEP that you want included in the expected set of MEPs for cross-check. |
| **Step 7** | **end** or **commit** | Saves configuration changes. |

| Command or Action | Purpose |
|---|---|
| **Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-xcheck)# commit | • When you use the **end** command, the system prompts you to commit changes:<br><br>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br><br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Other Options for a CFM Service

To configure other options for a CFM service, complete the following steps:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br>**Example:**<br><br>RP/0/RP0/CPU0:router# ethernet cfm | Enters the Ethernet Connectivity Fault Management (CFM) configuration mode. |
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:** | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified. |

| | Command or Action | Purpose |
|---|---|---|
| | RP/0/RP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1 | The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| Step 4 | **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* \| **down-meps** \| **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] \| [**string** *text*] \| [**number** *number*] \| [**vlan-id** *id-number*] \| [**vpn-id** *oui-vpnid*]]  **Example:**  RP/0/RP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1 | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created.  The **id** sets the short MA name. |
| Step 5 | **maximum-meps** *number*  **Example:**  RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# maximum-meps 1000 | (Optional) Configures the maximum number (2 to 8190) of MEPs across the network, which limits the number of peer MEPs recorded in the database. |
| Step 6 | **log** {**ais**\|**continuity-check errors**\|**continuity-check mep changes**\|**crosscheck errors**\|**efd**}  **Example:**  RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# log continuity-check errors | (Optional) Enables logging of certain types of events. |
| Step 7 | **end** or **commit**  **Example:**  RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# commit | Saves configuration changes.  • When you use the **end** command, the system prompts you to commit changes:  `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |

| Command or Action | Purpose |
|---|---|
| | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |
| | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring CFM MEPs

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| **Step 2** | **interface** {**HundredGigE | TenGigE**} *interface-path-id*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/1 | Type of Ethernet interface on which you want to create a MEP. Enter **HundredGigE** or **TenGigE** and the physical interface or virtual interface.<br><br>**Note**     • Use the **show interfaces** command to see a list of all interfaces currently configured on the router. |
| **Step 3** | **interface** {**HundredGigE | TenGigE | Bundle-Ether**} *interface-path-id***.***subinterface*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/1.1 | Type of Ethernet interface on which you want to create a MEP. Enter **HundredGigE, TenGigE,** or **Bundle-Ether** and the physical interface or virtual interface followed by the subinterface path ID.<br><br>Naming convention is *interface-path-id***.***subinterface*. The period in front of the subinterface value is required as part of the notation. |
| **Step 4** | **interface** {**HundredGigE | TenGigE**} *interface-path-id*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/1 | Type of Ethernet interface on which you want to create a MEP. Enter **HundredGigE** or **TenGigE** and the physical interface or virtual interface. |

| | Command or Action | Purpose |
|---|---|---|
| | | **Note** • Use the **show interfaces** command to see a list of all interfaces currently configured on the router. |
| **Step 5** | **ethernet cfm**<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if)# ethernet cfm | Enters interface Ethernet CFM configuration mode. |
| **Step 6** | **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if-cfm)# mep domain Dm1 service Sv1 mep-id 1 | Creates a maintenance end point (MEP) on an interface and enters interface CFM MEP configuration mode. |
| **Step 7** | **cos** *cos*<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if-cfm-mep)# cos 7 | (Optional) Configures the class of service (CoS) (from 0 to 7) for all CFM packets generated by the MEP on an interface. If not configured, the CoS is inherited from the Ethernet interface.<br><br>**Note** For Ethernet interfaces, the CoS is carried as a field in the VLAN tag. Therefore, CoS only applies to interfaces where packets are sent with VLAN tags. If the **cos (CFM)** command is executed for a MEP on an interface that does not have a VLAN encapsulation configured, it will be ignored. |
| **Step 8** | **end** or **commit**<br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if-cfm-mep)# commit | Saves configuration changes.<br><br>• When you use the **end** command, the system prompts you to commit changes:<br><br>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Y.1731 AIS

This section has the following step procedures:

## Configuring AIS in a CFM Domain Service

Use the following procedure to configure Alarm Indication Signal (AIS) transmission for a CFM domain service and configure AIS logging.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config)# ethernet cfm` | Enters Ethernet CFM global configuration mode. |
| **Step 3** | **domain** *name* **level** *level*<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-cfm)# domain D1 level 1` | Specifies the domain and domain level. |
| **Step 4** | **service** *name* **bridge group** *name* **bridge-domain** *name*<br><br>**Example:**<br><br>`RP/0/RP0/CPU0:router(config-cfm-dmn)# service S1 bridge group BG1 bridge-domain BD2` | Specifies the service, bridge group, and bridge domain. |

| | **Command or Action** | **Purpose** |
|---|---|---|
| **Step 5** | **service** *name* **xconnect group** *xconnect-group-name* **p2p** *xconnect-name* **Example:** RP/0/RP0/CPU0:router(config-cfm-dmn)# service S1 xconnect group XG1 p2p X2 | Specifies the service and cross-connect group and name. |
| **Step 6** | **ais transmission** [**interval** {**1s**|**1m**}][**cos** *cos*] **Example:** RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7 | Configures Alarm Indication Signal (AIS) transmission for a Connectivity Fault Management (CFM) domain service. |
| **Step 7** | **log ais** **Example:** RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# log ais | Configures AIS logging for a Connectivity Fault Management (CFM) domain service to indicate when AIS or LCK packets are received. |
| **Step 8** | **no domain** *name***level** *level* **Example:** RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# no domain D1 level 1 | Disables the domain and domain level. |
| **Step 9** | **end** or **commit** **Example:** RP/0/RP0/CPU0:router(config-sla-prof-stat-cfg)# commit | Saves configuration changes. <ul><li>When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`</li><li>Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</li><li>Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</li><li>Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.</li><li>Use the **commit** command to save the configuration changes to the running</li></ul> |

| Command or Action | Purpose |
|---|---|
| | configuration file and remain within the configuration session. |

## Configuring AIS on a CFM Interface

To configure AIS on a CFM interface, perform the following steps:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| Step 2 | **interface gigabitethernet** *interface-path-id*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# interface TenGigE 0/0/0/2 | Enters interface configuration mode. |
| Step 3 | **ethernet cfm**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# ethernet cfm | Enters Ethernet CFM interface configuration mode. |
| Step 4 | **ais transmission up interval 1m cos** *cos*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-if-cfm)# ais transmission up interval 1m cos 7 | Configures Alarm Indication Signal (AIS) transmission on a Connectivity Fault Management (CFM) interface. |
| Step 5 | **end** or **commit**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-sla-prof-stat-cfg)# commit | Saves configuration changes.<br><br>• When you issue the **end** command, the system prompts you to commit changes:<br><br>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode |

| | Command or Action | Purpose |
|---|---|---|
| | | without committing the configuration changes. |
| | | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Flexible VLAN Tagging for CFM

Use this procedure to set the number of tags in CFM packets in a CFM domain service.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| Step 2 | **ethernet cfm**<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config)# ethernet cfm | Enters Ethernet CFM global configuration mode. |
| Step 3 | **domain** *name* **level** *level*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm)# domain D1 level 1 | Specifies the domain and domain level. |
| Step 4 | **service** *name* **bridge group** *name* **bridge-domain** *name*<br><br>**Example:**<br><br>RP/0/RP0/CPU0:router(config-cfm-dmn)# service S2 bridge group BG1 bridge-domain BD2 | Specifies the service, bridge group, and bridge domain. |
| Step 5 | **tags** *number*<br><br>**Example:** | Specifies the number of tags in CFM packets. Currently, the only valid value is 1. |

| | Command or Action | Purpose |
|---|---|---|
| | ```RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# tags 1``` | |
| Step 6 | **end** or **commit**<br><br>**Example:**<br><br>```RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# commit``` | Saves configuration changes.<br><br>• When you issue the **end** command, the system prompts you to commit changes:<br><br>```Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:```<br><br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br><br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Verifying the CFM Configuration

To verify the CFM configuration, use one or more of the following commands:

| | |
|---|---|
| **show ethernet cfm configuration-errors** [**domain** *domain-name*] [**interface** *interface-path-id* ] | Displays information about errors that are preventing configured CFM operations from becoming active, as well as any warnings that have occurred. |
| **show ethernet cfm local maintenance-points domain** *name* [**service** *name]* \| **interface** *type interface-path-id*] [**mep** \| **mip**] | Displays a list of local maintenance points. |

✎

**Note**  After you configure CFM, the error message, *cfmd[317]: %L2-CFM-5-CCM_ERROR_CCMS_MISSED : Some received CCMs have not been counted by the CCM error counters*, may display. This error message does not have any functional impact and does not require any action from you.

## Troubleshooting Tips

To troubleshoot problems within the CFM network, perform these steps:

**Procedure**

**Step 1**    To verify connectivity to a problematic MEP, use the **ping ethernet cfm** command as shown in this example:

```
RP/0/RP0/CPU0:router# ping ethernet cfm domain D1 service S1 mep-id 16 source
interface TenGigE 0/0/0/1

Type escape sequence to abort.
Sending 5 CFM Loopbacks, timeout is 2 seconds -
Domain foo (level 2), Service foo
Source: MEP ID 1, interface TenGigE0/0/0/1
Target: 0001.0002.0003 (MEP ID 16):
  Running (5s) ...
Success rate is 60.0 percent (3/5), round-trip min/avg/max = 1251/1349/1402 ms
Out-of-sequence: 0.0 percent (0/3)
Bad data: 0.0 percent (0/3)
Received packet rate: 1.4 pps
```

**Step 2**    If the results of the **ping ethernet cfm** command show a problem with connectivity to the peer MEP, use the **traceroute ethernet cfm** command to help further isolate the location of the problem as shown in the following example:

```
RP/0/RP0/CPU0:router# traceroute ethernet cfm domain D1 service S1 mep-id 16
 source interface TenGigE 0/0/0/2

Traceroutes in domain D1 (level 4), service S1
Source: MEP-ID 1, interface TenGigE0/0/0/2
================================================================================
Traceroute at 2009-05-18 12:09:10 to 0001.0203.0402,
TTL 64, Trans ID 2:

Hop Hostname/Last           Ingress MAC/name        Egress MAC/Name         Relay
--- ---------------------- ---------------------- ---------------------- -----
  1 ios                     0001.0203.0400 [Down]                          FDB
    0000-0001.0203.0400     TenGigE0/0/0/2
  2 abc                                             0001.0203.0401 [Ok]    FDB
    ios                                             Not present
  3 bcd                     0001.0203.0402 [Ok]                            Hit
    abc                     TenGigE0/0
Replies dropped: 0
```

If the target was a MEP, verify that the last hop shows "Hit" in the Relay field to confirm connectivity to the peer MEP.

If the Relay field contains "MPDB" for any of the hops, then the target MAC address was not found in the bridge MAC learning table at that hop, and the result is relying on CCM learning. This result can occur under normal conditions, but it can also indicate a problem. If you used the **ping ethernet cfm** command before using the **traceroute ethernet cfm** command, then the MAC address should have been learned. If "MPDB" is appearing in that case, then this indicates a problem at that point in the network.

# Configuration Examples for Ethernet OAM

This section provides the following configuration examples:

## Configuration Examples for EOAM Interfaces

This section provides the following configuration examples:

### Configuring an Ethernet OAM Profile Globally: Example

This example shows how to configure an Ethernet OAM profile globally:

```
configure terminal
 ethernet oam profile Profile_1
  link-monitor
   symbol-period window 60000
   symbol-period threshold ppm low 10000000 high 60000000
   frame window 60
   frame threshold ppm low 10000000 high 60000000
   frame-period window 60000
   frame-period threshold ppm low 100 high 12000000
   frame-seconds window 900000
   frame-seconds threshold low 3 high 900
   exit
  mib-retrieval
  connection timeout 30
  require-remote mode active
  require-remote mib-retrieval
  action dying-gasp error-disable-interface
  action critical-event error-disable-interface
  action discovery-timeout error-disable-interface
  action session-down error-disable-interface
  action capabilities-conflict error-disable-interface
  action wiring-conflict error-disable-interface
  action remote-loopback error-disable-interface
  commit
```

### Configuring Ethernet OAM Features on an Individual Interface: Example

This example shows how to configure Ethernet OAM features on an individual interface:

```
configure terminal
 interface TenGigE 0/0/0/0
  ethernet oam
   link-monitor
    symbol-period window 60000
    symbol-period threshold ppm low 10000000 high 60000000
    frame window 60
    frame threshold ppm low 10000000 high 60000000
    frame-period window 60000
    frame-period threshold ppm low 100 high 12000000
    frame-seconds window 900000
    frame-seconds threshold low 3 high 900
    exit
   mib-retrieval
   connection timeout 30
   require-remote mode active
   require-remote mib-retrieval
   action link-fault  error-disable-interface
   action dying-gasp error-disable-interface
```

```
action critical-event error-disable-interface
action discovery-timeout error-disable-interface
action session-down error-disable-interface
action capabilities-conflict error-disable-interface
action wiring-conflict error-disable-interface
action remote-loopback error-disable-interface
commit
```

## Configuring Ethernet OAM Features to Override the Profile on an Individual Interface: Example

This example shows the configuration of Ethernet OAM features in a profile followed by an override of that configuration on an interface:

```
configure terminal
 ethernet oam profile Profile_1
  mode passive
  action dying-gasp disable
  action critical-event disable
  action discovery-timeout disable
  action session-up disable
  action session-down disable
  action capabilities-conflict disable
  action wiring-conflict disable
  action remote-loopback disable
  action uni-directional link-fault error-disable-interface
  commit

configure terminal
 interface TenGigE 0/0/0/0
  ethernet oam
   profile Profile_1
    mode active
    action dying-gasp log
    action critical-event log
    action discovery-timeout log
    action session-up log
    action session-down log
    action capabilities-conflict log
    action wiring-conflict log
    action remote-loopback log
    action uni-directional link-fault log
    uni-directional link-fault detection
    commit
```

## Clearing Ethernet OAM Statistics on an Interface: Example

This example shows how to clear Ethernet OAM statistics on an interface:

```
RP/0/RP0/CPU0:router# clear ethernet oam statistics interface gigabitethernet 0/0/0/1
```

## Enabling SNMP Server Traps on a Router: Example

This example shows how to enable SNMP server traps on a router:

```
configure terminal
 snmp-server traps ethernet oam events
```

# Configuration Examples for Ethernet CFM

This section includes the following examples:

## Ethernet CFM Domain Configuration: Example

This example shows how to configure a basic domain for Ethernet CFM:

```
configure
 ethernet cfm
  traceroute cache hold-time 1 size 3000
  domain Domain_One level 1 id string D1
  commit
```

## Ethernet CFM Service Configuration: Example

This example shows how to create a service for an Ethernet CFM domain:

```
  service Bridge_Service bridge group BD1 bridge-domain B1
  service Cross_Connect_1 xconnect group XG1 p2p X1
  commit
```

## Flexible Tagging for an Ethernet CFM Service Configuration: Example

This example shows how to set the number of tags in CFM packets from down MEPs in a CFM domain service:

```
configure
 ethernet cfm
  domain D1 level 1
   service S2 bridge group BG1 bridge-domain BD2
    tags 1
    commit
```

## Continuity Check for an Ethernet CFM Service Configuration: Example

This example shows how to configure continuity-check options for an Ethernet CFM service:

```
  continuity-check archive hold-time 100
  continuity-check loss auto-traceroute
  continuity-check interval 100ms loss-threshold 10
  commit
```

## MIP Creation for an Ethernet CFM Service Configuration: Example

This example shows how to enable MIP auto-creation for an Ethernet CFM service:

```
RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# mip auto-create all
RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# commit
```

## Cross-check for an Ethernet CFM Service Configuration: Example

This example shows how to configure cross-check for MEPs in an Ethernet CFM service:

```
  mep crosscheck
   mep-id 10
   mep-id 20
   commit
```

## Other Ethernet CFM Service Parameter Configuration: Example

This example shows how to configure other Ethernet CFM service options:

```
  maximum-meps 4000
  log continuity-check errors
  commit
  exit
 exit
exit
```

## MEP Configuration: Example

This example shows how to configure a MEP for Ethernet CFM on an interface:

```
interface TenGigE 0/0/0/1
 ethernet cfm
 mep domain Dm1 service Sv1 mep-id 1
 commit
```

## Ethernet CFM Show Command: Examples

These examples show how to verify the configuration of Ethernet Connectivity Fault Management (CFM):

### Example 1

This example shows how to display all the maintenance points that have been created on an interface:

```
RP/0/RP0/CPU0:router# show ethernet cfm local maintenance-points

Domain/Level         Service             Interface         Type   ID   MAC
-------------------- ------------------- ----------------- ------ ---- --------
fig/5                bay                 Gi0/10/0/12       Dn MEP  2 44:55:66
fig/5                bay                 Gi0/0/1/0         MIP       55:66:77
fred/3               barney              Gi0/1/0/0         Dn MEP  5 66:77:88!
```

### Example 2

This example shows how to display all the CFM configuration errors on all domains:

```
RP/0/RP0/CPU0:router# show ethernet cfm configuration-errors

Domain fig (level 5), Service bay
* MIP creation configured using bridge-domain blort, but bridge-domain blort does not exist.

 * An Up MEP is configured for this domain on interface TenGigE0/0/0/3 and an Up MEP is
also configured for domain blort, which is at the same level (5).
 * A MEP is configured on interface TenGigE0/0/0/1 for this domain/service, which has CC
interval 100ms, but the lowest interval supported on that interface is 1s
```

### Example 3

This example shows how to display operational state for local maintenance end points (MEPs):

```
RP/0/RP0/CPU0:router# show ethernet cfm local meps

A - AIS received              I - Wrong interval
R - Remote Defect received    V - Wrong Level
L - Loop (our MAC received)   T - Timed out (archived)
C - Config (our ID received)  M - Missing (cross-check)
X - Cross-connect (wrong MAID) U - Unexpected (cross-check)
P - Peer port down
```

```
Domain foo (level 6), Service bar
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
  100 Gi1/1/0/1 (Up)          Up    0/0    N  A       L7

Domain fred (level 5), Service barney
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
    2 Gi0/1/0/0 (Up)          Up    3/2    Y  RPC     L6
Domain foo (level 6), Service bar
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
  100 Gi1/1/0/1 (Up)          Up    0/0    N  A

Domain fred (level 5), Service barney
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
    2 Gi0/1/0/0 (Up)          Up    3/2    Y  RPC
```

### Example 4

This example shows how to display operational state of other maintenance end points (MEPs) detected by a local MEP:

```
RP/0/RP0/CPU0:router# show ethernet cfm peer meps

Flags:
 > - Ok                           I - Wrong interval
 R - Remote Defect received       V - Wrong level
 L - Loop (our MAC received)      T - Timed out
 C - Config (our ID received)     M - Missing (cross-check)
 X - Cross-connect (wrong MAID)   U - Unexpected (cross-check)

Domain fred (level 7), Service barney
Down MEP on TenGigE0/0/0/1, MEP-ID 2
================================================================================
St    ID MAC address     Port    Up/Downtime   CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- --------- ------ ----- -----
 >    1 0011.2233.4455 Up       00:00:01       1234      0     0     0
R>    4 4455.6677.8899 Up       1d 03:04       3456      0   234     0
L     2 1122.3344.5566 Up       3w 1d 6h       3254      0     0  3254
C     2 7788.9900.1122 Test     00:13          2345      6    20  2345
X     3 2233.4455.6677 Up       00:23            30      0     0    30
I     3 3344.5566.7788 Down     00:34         12345      0   300  1234
V     3 8899.0011.2233 Blocked 00:35            45      0     0    45
 T    5 5566.7788.9900          00:56            20      0     0     0
M     6                                           0      0     0     0
U>    7 6677.8899.0011 Up       00:02           456      0     0     0

Domain fred (level 7), Service fig
Down MEP on TenGigE0/0/0/12, MEP-ID 3
================================================================================
St    ID MAC address     Port    Up/Downtime  CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- -------- ------ ----- -----
 >    1 9900.1122.3344 Up       03:45          4321      0     0     0
```

### Example 5

This example shows how to display operational state of other maintenance end points (MEPs) detected by a local MEP with details:

```
RP/0/RP0/CPU0:router#  show ethernet cfm peer meps detail
Domain dom3 (level 5), Service ser3
Down MEP on TenGigE0/0/0/1 MEP-ID 1
================================================================================
Peer MEP-ID 10, MAC 0001.0203.0403
   CFM state: Wrong level, for 00:01:34
   Port state: Up
   CCM defects detected:    V - Wrong Level
   CCMs received: 5
     Out-of-sequence:            0
     Remote Defect received:     5
     Wrong Level:                0
     Cross-connect (wrong MAID): 0
     Wrong Interval:             5
     Loop (our MAC received):    0
     Config (our ID received):   0
Last CCM received 00:00:06 ago:
     Level: 4, Version: 0, Interval: 1min
     Sequence number: 5, MEP-ID: 10
     MAID: String: dom3, String: ser3
     Port status: Up, Interface status: Up


Domain dom4 (level 2), Service ser4
Down MEP on TenGigE0/0/0/2 MEP-ID 1
================================================================================
Peer MEP-ID 20, MAC 0001.0203.0402
   CFM state: Ok, for 00:00:04
   Port state: Up
   CCMs received: 7
     Out-of-sequence:            1
     Remote Defect received:     0
     Wrong Level:                0
     Cross-connect (wrong MAID): 0
     Wrong Interval:             0
     Loop (our MAC received):    0
  Config (our ID received):    0
Last CCM received 00:00:04 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 20
     MAID: String: dom4, String: ser4
     Chassis ID: Local: ios; Management address: 'Not specified'
     Port status: Up, Interface status: Up

Peer MEP-ID 21, MAC 0001.0203.0403
   CFM state: Ok, for 00:00:05
   Port state: Up
   CCMs received: 6
     Out-of-sequence:            0
     Remote Defect received:     0
     Wrong Level:                0
     Cross-connect (wrong MAID): 0
     Wrong Interval:             0
     Loop (our MAC received):    0
     Config (our ID received):   0
Last CCM received 00:00:05 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 21
     MAID: String: dom4, String: ser4
     Port status: Up, Interface status: Up


Peer MEP-ID 601, MAC 0001.0203.0402
```

```
        CFM state: Timed Out (Standby), for 00:15:14, RDI received
        Port state: Down
        CCM defects detected:    Defects below ignored on local standby MEP
                                 I - Wrong Interval
                                 R - Remote Defect received
                                 T - Timed Out
                                 P - Peer port down
      CCMs received: 2
        Out-of-sequence:              0
        Remote Defect received:       2
        Wrong Level:                  0

        Wrong Interval:               2
        Loop (our MAC received):      0
        Config (our ID received):     0
      Last CCM received 00:15:49 ago:
        Level: 2, Version: 0, Interval: 10s
        Sequence number: 1, MEP-ID: 600
        MAID: DNS-like: dom5, String: ser5
        Chassis ID: Local: ios; Management address: 'Not specified'
        Port status: Up, Interface status: Down
```

## AIS for CFM Configuration: Examples

### Example 1

This example shows how to configure Alarm Indication Signal (AIS) transmission for a CFM domain service:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ethernet cfm
RP/0/RP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RP0/CPU0:router(config-cfm-dmn)# service S1 bridge group BG1 bridge-domain BD2
RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7

RP/0/RP0/CPU0:routerconfigure
RP/0/RP0/CPU0:router(config)# ethernet cfm
RP/0/RP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RP0/CPU0:router(config-cfm-dmn)# service Cross_Connect_1 xconnect group XG1 p2p X1
RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7
```

### Example 2

This example shows how to configure AIS logging for a Connectivity Fault Management (CFM) domain service to indicate when AIS or LCK packets are received:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ethernet cfm
RP/0/RP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RP0/CPU0:router(config-cfm-dmn)# service S2 bridge group BG1 bridge-domain BD2
RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# log ais

RP/0/RP0/CPU0:routerconfigure
RP/0/RP0/CPU0:router(config)# ethernet cfm
RP/0/RP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RP0/CPU0:router(config-cfm-dmn)# service Cross_Connect_1 xconnect group XG1 p2p X1
RP/0/RP0/CPU0:router(config-cfm-dmn-svc)# log ais
```

This example shows how to configure AIS transmission on a CFM interface.

```
RP/0/RP0/CPU0:router# configure
```

```
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/RP0/CPU0:router(config-if)# ethernet cfm
RP/0/RP0/CPU0:router(config-if-cfm)# ais transmission up interval 1m cos 7
```

## AIS for CFM Show Commands: Examples

This section includes the following examples:

### show ethernet cfm interfaces ais Command: Example

This example shows how to display the information published in the Interface AIS table:

```
RP/0/RP0/CPU0:router# show ethernet cfm interfaces ais

Defects (from at least one peer MEP):
 A - AIS received              I - Wrong interval
 R - Remote Defect received    V - Wrong Level
 L - Loop (our MAC received)   T - Timed out (archived)
 C - Config (our ID received)  M - Missing (cross-check)
 X - Cross-connect (wrong MAID) U - Unexpected (cross-check)
 P - Peer port down            D - Local port down

                          Trigger              Transmission
                      AIS --------- Via   --------------------------
Interface (State)     Dir L Defects Levels L Int Last started Packets
---------------------- --- - ------- ------- - --- ------------ --------
TenGigE0/0/0/0 (Up)    Dn  5 RPC     6       7 1s  01:32:56 ago    5576
TenGigE0/0/0/0 (Up)    Up  0 M       2,3     5 1s  00:16:23 ago     983
TenGigE0/0/0/1 (Dn)    Up    D               7 60s 01:02:44 ago    3764
TenGigE0/0/0/2 (Up)    Dn  0 RX      1!
```

### show ethernet cfm local meps Command: Examples

#### Example 1: Default

This example shows how to display statistics for local maintenance end points (MEPs):

```
RP/0/RP0/CPU0:router# show ethernet cfm local meps

 A - AIS received              I - Wrong interval
 R - Remote Defect received    V - Wrong Level
 L - Loop (our MAC received)   T - Timed out (archived)
 C - Config (our ID received)  M - Missing (cross-check)
 X - Cross-connect (wrong MAID) U - Unexpected (cross-check)
 P - Peer port down

Domain foo (level 6), Service bar
   ID Interface (State)       Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
  100 Gi1/1/0/1 (Up)          Up    0/0     N  A         7

Domain fred (level 5), Service barney
   ID Interface (State)       Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
    2 Gi0/1/0/0 (Up)          Up    3/2     Y  RPC       6
```

#### Example 2: Domain Service

This example shows how to display statistics for MEPs in a domain service:

```
RP/0/RP0/CPU0:router# show ethernet cfm local meps domain foo service bar detail

Domain foo (level 6), Service bar
Down MEP on TenGigE0/0/0/1, MEP-ID 100
================================================================================
  Interface state: Up     MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)

Domain fred (level 5), Service barney
Down MEP on TenGigE0/0/0/1, MEP-ID 2
================================================================================
  Interface state: Up     MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: Yes)
  CCM defects detected:    R - Remote Defect received
                           P - Peer port down
                           C - Config (our ID received)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             Yes (to higher MEP, started 01:32:56 ago)
  Receiving AIS:           No
```

### Example 4: Detail

This example shows how to display detailed statistics for MEPs in a domain service:

```
RP/0/RP0/CPU0:router# show ethernet cfm local meps detail

Domain foo (level 6), Service bar
Down MEP on TenGigE0/0/0/1, MEP-ID 100
================================================================================
  Interface state: Up     MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)

Domain fred (level 5), Service barney
Down MEP on TenGigE0/0/0/1, MEP-ID 2
================================================================================
  Interface state: Up     MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: Yes)
  CCM defects detected:    R - Remote Defect received
                           P - Peer port down
                           C - Config (our ID received)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             Yes (to higher MEP, started 01:32:56 ago)
  Receiving AIS:           No
```

## show ethernet cfm local meps detail Command: Example

Use the **show ethernet cfm local meps detail** command to display MEP-related EFD status information. This example shows that EFD is triggered for MEP-ID 100:

```
RP/0/RP0/CPU0:router# show ethernet cfm local meps detail

Domain foo (level 6), Service bar
Down MEP on TenGigE0/0/0/1, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 2 missing, 0 unexpected

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)
  EFD triggered:           Yes

Domain fred (level 5), Service barney
Down MEP on TenGigE0/0/0/1, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: No)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             No
  Receiving AIS:           No
  EFD triggered:           No
```

**Note**    You can also verify that EFD has been triggered on an interface using the **show interfaces** and **show interfaces brief** commands. When an EFD trigger has occurred, these commands will show the interface status as *up* and the line protocol state as *down*.

# CFM Over Bundles

CFM over bundle supports the following:

- CFM Maintenance Points—Up Maintenance-association End Points (MEP), Down MEP, and MIP, which includes L2 bundle main and sub-interfaces.

- CCM interval of 100 microsecond, 1second, 10 seconds, and 1 minute. CCM interval of 10 minutes is supported only in the versions earlier than IOS XR 7.3.2.

- RP OIR/VM reload, without impacting learned CFM peer MEPs.

- Process restart without impacting CFM sessions.

- CFM MEPs on bundle interfaces as software-offloaded-MEPs with all possible rewrite and encapsulation combinations supported by L2 sub-interfaces.

- CCM learning on MIP over bundle interfaces. CCM database learning supports investigation of one CCM out of 50 that goes over MIP.

- Static and dynamic MEPs.

### Restrictions for Configuration of CFM on Bundles

Following are the restrictions for configuring CFM over bundle member interfaces:

- Only Layer 2 bundle Ethernet interfaces and sub-interfaces are supported except for those matching the VLAN tag `any`.

- CCM interval of 3.3 milliseconds and 10 milliseconds are not supported.

- CCM interval of 10 minutes is not supported from IOS XR 7.3.2.

- Supports 5000 pps rates of CCM traffic for bundle interfaces.

- Ethernet CFM is not supported with MEP that are configured on default and untagged encapsulated sub-interfaces that are part of a single physical interface.

# Unidirectional Link Detection Protocol

Unidirectional Link Detection (UDLD) is a single-hop physical link protocol for monitoring an ethernet link, including both point-to-point and shared media links. This is a Cisco-proprietary protocol to detect link problems, which are not detected at the physical link layer. This protocol is specifically targeted at possible wiring errors, when using unbundled fiber links, where there can be a mismatch between the transmitting and receiving connections of a port.

### Limitations

- UDLD is not tunneled through L2VPN like other slow protocols.

- UDLD must not be enabled on a Switched Port Analyzer (SPAN) source or a destination port.

- UDLD must not be enabled on a port that acts as a source or destination port for SPAN.

# UDLD Operation

UDLD works by exchanging protocol packets between the neighboring devices. In order for UDLD to work, both devices on the link must support UDLD and have it enabled on respective ports.

UDLD sends an initial PROBE message on the ports where it is configured. Once UDLD receives a PROBE message, it sends periodic ECHO (hello) messages. Both messages identify the sender and its port, and also contain some information about the operating parameters of the protocol on that port. They also contain the device and port identifiers on the port for any neighbor devices that the local device has heard from. Similarly, each device gets to know where it is connected and where its neighbors are connected. This information can then be used to detect faults and miswiring conditions.

The protocol operates an aging mechanism where information from neighbors that is not periodically refreshed is eventually timed out. This mechanism can also be used to detect fault.

A FLUSH message is used to indicate that UDLD is disabled on a port, which causes the peers to remove the local device from their neighbor cache to prevent a time out.

If a problem is detected, UDLD disables the affected interface and also notifies the user. This is to avoid further network problems beyond traffic loss, such as loops which are not detected or prevented by Spanning Tree Protocol (STP).

# Types of Fault Detection

UDLD can detect these types of faults:

- Transmit faults — These are cases where there is a failure in transmitting packets from the local port to the peer device, but packets are being received from the peer. These faults are caused by failure of the physical link (where notification at layer 1 of unidirectional link faults is not supported by the media) as well as packet path faults on the local or peer device.

- Miswiring faults — These are cases where the receiving and transmitting sides of a port on the local device are connected to different peer ports (on the same device or on different devices). This can occur when using unbundled fibers to connect fiber optic ports.

- Loopback faults — These are cases where the receiving and transmitting sides of a port are connected to each other, creating a loopback condition. This can be an intentional mode of operation, for certain types of testing, but UDLD must not be used in these cases.

- Receive faults — The protocol includes a heartbeat signal that is transmitted at a negotiated periodic interval to the peer device. Missed heartbeats can therefore be used to detect failures on the receiving side of the link (where they do not result in interface state changes). These could be caused by a unidirectional link with a failure only affecting the receiving side, or by a link which has developed a bidirectional fault. This detection depends on reliable, regular packet transmission by the peer device. For this reason, the UDLD protocol has two (configurable) modes of operation which determine the behavior on a heartbeat timeout. These modes are described in the section UDLD Modes of Operation, on page 55.

# UDLD Modes of Operation

UDLD can operate in these modes:

- **Normal mode**: In this mode, if a `Receive Fault` is detected, the user is informed and no further action is taken.

- **Aggressive mode**: In this mode, if a `Receive Fault` is detected, the user is informed and the affected port is disabled.

**Note** The difference of behavior between normal and aggressive modes is only seen in case of neighbor timeout. In all other cases, irrespective of the normal or aggressive mode, the system error disables a link once a unidirectional link is detected.

# UDLD Aging Mechanism

This is a scenario that happens in a `Receive Fault` condition. Aging of UDLD information happens when the port that runs UDLD does not receive UDLD packets from the neighbor port for a duration of the hold time. The hold time for the port is dictated by the remote port and depends on the message interval at the remote side. The shorter the message interval, the shorter is the hold time and the faster the detection of the fault. The hold time is three times the message interval in Cisco IOS XR Software.

UDLD information can age out due to the high error rate on the port caused by a physical issue or duplex mismatch. Packet drops due to age out does not mean that the link is unidirectional. UDLD in normal mode does not disable such link.

It is important to choose the right message interval in order to ensure proper detection time. The message interval should be fast enough to detect the unidirectional link before the forwarding loop is created. The default message interval is 60 seconds. The detection time is equal to approximately three times the message interval. So, when using default UDLD timers, UDLD does not timeout the link faster than the STP aging time.

# State Machines

UDLD uses two types of finite state machines (FSMs), generally referred as state machines. The `Main FSM` deals with all the phases of operation of the protocol while the `Detection FSM` handles only the phases that determine the status of a port.

# Main FSM

The Main FSM can be in one of these states:

- **Init**: Protocol is initializing.

- **UDLD inactive**: Port is down or UDLD is disabled.

- **Linkup**: Port is up and running, and UDLD is in the process of detecting a neighbor.

- **Detection**: A hello message from a new neighbor is received and the Detection FSM determines the status of the port.

- **Advertisement**: The Detection FSM concludes that the port is operating correctly, periodic hello messages will continue to be sent and monitored from neighbors.

- **Port shutdown**: The Detection FSM detected a fault, or all neighbors were timed out in Aggressive mode, and as a result, the port is disabled.

# Detection FSM

The Detection FSM can be in one of these states:

- **Unknown**: Detection has not yet been performed or UDLD has been disabled.

- **Unidirectional detected**: A unidirectional link condition has been detected because a neighbor does not see the local device. The port will be disabled.

- **Tx/Rx loop**: A loopback condition has been detected by receiving a TLV with the ports own identifiers. The port will be disabled.

- **Neighbor mismatch**: A miswiring condition has been detected in which a neighbor can identify other devices than those the local device can see. The port will be disabled.

- **Bidirectional detected**: UDLD hello messages are exchanged successfully in both directions. The port is operating correctly.

# Y.1731 Performance Monitoring

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. This is specified by the ITU-T Y-1731 standard and interpreted by the Metro Ethernet Forum (MEF) standards group.

The router supports the following:

- Delay Measurement (DM)

- Synthetic Loss Measurement (SLM)

# Two-Way Delay Measurement for Scalability

Use the Ethernet frame delay measurement to measure frame delay and frame delay variations. The system measures the Ethernet frame delay by using the Delay Measurement Message (DMM) method.

### Restrictions for Configuring Two-Way Delay Measurement

Follow the guidelines and restrictions listed here when you configure two-way delay measurement:

- Y.1731 PM is not supported for One-Way DMM in release prior to XR Release 6.6.25.

## Configuring Two-Way Delay Measurement

Perform the following steps to configure two-way delay measurement:

RP/0/RP0/CPU0:router **(config)** # **ethernet sla**

```
profile DMM type cfm-delay-measurement
  probe
   send burst every 5 seconds packet count 5 interval 1 seconds
  !
  schedule
   every 1 minutes for 40 seconds
  !
  statistics
   measure round-trip-delay
    buckets size 1 probes
    buckets archive 5
   !
   measure round-trip-jitter
    buckets size 1 probes
    buckets archive 1
   !
!
```

```
!
!
interface TenGigE0/0/0/10.1 l2transport
encapsulation dot1q 1
ethernet cfm
  mep domain DOWN0 service s10 mep-id 2001
   sla operation profile DMM target mep-id 6001
  !
```

### Configuring an On-Demand Ethernet SLA Operation for CFM Delay Measurement

To configure an on-demand Ethernet SLA operation for CFM delay measurement, use this command in privileged EXEC configuration mode:

RP/0/RP0/CPU0:router (config) #

```
ethernet sla on-demand operation type cfm-synthetic-loss-measurement probe domain D1 source
 interface TenGigE 0/6/1/0 target mac-address 2.3.4
```

### Running Configuration

```
RP/0/RP0/CPU0:router# show ethernet cfm peer meps
Mon Sep 11 12:09:44.534 UTC
Flags:
> - Ok                       I - Wrong interval
R - Remote Defect received    V - Wrong level
L - Loop (our MAC received)   T - Timed out
C - Config (our ID received)  M - Missing (cross-check)
X - Cross-connect (wrong MAID) U - Unexpected (cross-check)
* - Multiple errors received  S - Standby

Domain UP6 (level 6), Service s6
Up MEP on FortyGigE0/0/1/2.1 MEP-ID 1
================================================================================
St   ID MAC Address    Port    Up/Downtime  CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- --------- ------ ----- -----
 >  4001 70e4.227c.2865 Up      00:01:27           0      0     0     0

Domain DOWN0 (level 0), Service s10
Down MEP on TenGigE0/0/0/10.1 MEP-ID 2001
================================================================================
St   ID MAC Address    Port    Up/Downtime  CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- --------- ------ ----- -----
 >  6001 70e4.227c.287a Up      00:02:11           0      0     0     0
RP/0/RP0/CPU0:router#
RP/0/RP0/CPU0:router# show running-config
Mon Sep 11 12:10:18.467 UTC
Building configuration...
!! IOS XR Configuration version = 6.4.1.14
!! Last configuration change at Mon Sep 11 12:08:16 2017 by root
!
logging console disable
telnet vrf default ipv4 server max-servers 10
username root
group root-lr
group cisco-support
secret 5 $1$QJT3$94M5/wK5J0v/lpAu/wz31/
!
line console
exec-timeout 0 0
!
ethernet cfm
domain UP6 level 6 id null
```

```
     service s6 xconnect group g1 p2p p1 id number 6
      mip auto-create all ccm-learning
      continuity-check interval 1s
      mep crosscheck
       mep-id 4001
       !
      !
 !
domain DOWN0 level 0 id null
 service s10 down-meps id number 10
      continuity-check interval 1s
      mep crosscheck
       mep-id 6001
       !
      !
 !
 !
profile DMM type cfm-delay-measurement
  probe
   send burst every 5 seconds packet count 5 interval 1 seconds
   !
  schedule
   every 1 minutes for 40 seconds
   !
  statistics
   measure round-trip-delay
    buckets size 1 probes
    buckets archive 5
    !
   measure round-trip-jitter
    buckets size 1 probes
    buckets archive 1
    !
interface MgmtEth0/RP0/CPU0/0
shutdown
!
interface TenGigE0/0/0/0
shutdown
!
interface TenGigE0/0/0/1
shutdown
!
interface TenGigE0/0/0/2
shutdown
!
interface TenGigE0/0/0/3
shutdown
!
interface TenGigE0/0/0/4
shutdown
!
interface TenGigE0/0/0/5
shutdown
!
interface TenGigE0/0/0/6
shutdown
!
interface TenGigE0/0/0/7
shutdown
!
interface TenGigE0/0/0/8
shutdown
!
interface TenGigE0/0/0/9
```

```
shutdown
!
interface TenGigE0/0/0/10.1 l2transport
encapsulation dot1q 1
ethernet cfm
  mep domain DOWN0 service s10 mep-id 2001
   sla operation profile DMM target mep-id 6001
   sla operation profile test-slm target mep-id 6001
  !
!
!
interface TenGigE0/0/0/11
shutdown
!
interface TenGigE0/0/0/12
shutdown
!
interface TenGigE0/0/0/13
shutdown
!
interface TenGigE0/0/0/14
shutdown
!
interface TenGigE0/0/0/15
shutdown
!
interface TenGigE0/0/0/16
shutdown
!
interface TenGigE0/0/0/17
shutdown
!
interface TenGigE0/0/0/18
shutdown
!
interface TenGigE0/0/0/19
shutdown
!
interface TenGigE0/0/0/20
shutdown
!
interface TenGigE0/0/0/21
shutdown
!
interface TenGigE0/0/0/22
shutdown
!
interface TenGigE0/0/0/23
shutdown
!
interface TenGigE0/0/0/24
shutdown
!
interface TenGigE0/0/0/25
shutdown
!
interface TenGigE0/0/0/26
shutdown
!
interface TenGigE0/0/0/27
shutdown
!
interface TenGigE0/0/0/28
shutdown
```

```
!
interface TenGigE0/0/0/29
shutdown
!
interface TenGigE0/0/0/30
shutdown
!
!
interface TenGigE0/0/0/31
 shutdown
!
interface TenGigE0/0/0/32
 shutdown
!
interface TenGigE0/0/0/33
 shutdown
!
interface TenGigE0/0/0/34
 shutdown
!
interface TenGigE0/0/0/35
 shutdown
!
interface TenGigE0/0/0/36
 shutdown
!
interface TenGigE0/0/0/37
 shutdown
!
interface TenGigE0/0/0/38
 shutdown
!
interface TenGigE0/0/0/39
 shutdown
!
interface TenGigE0/0/1/0/1
 shutdown
!
interface TenGigE0/0/1/0/2
 shutdown
!
interface TenGigE0/0/1/0/3
 shutdown
!
controller Optics0/0/1/0
 breakout 4x10
!
interface HundredGigE0/0/1/1
 shutdown
!
interface FortyGigE0/0/1/2.1 l2transport
 encapsulation dot1q 1
 ethernet cfm
  mep domain UP6 service s6 mep-id 1
    sla operation profile DMM target mep-id 6001
    sla operation profile test-slm target mep-id 6001
  !
 !
!
l2vpn
 xconnect group g1
  p2p p1
    interface TenGigE0/0/0/10.1
    interface FortyGigE0/0/1/2.1
```

```
   !
  !
 !
 end
```

## Verification

```
One-way Delay (Source->Dest)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 10
    Min: 1912765.961ms; Max: 1912765.961ms; Mean: 1912765.961ms; StdDev: -2147483.648ms


One-way Delay (Dest->Source)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 10
    Min: -1912765.952ms; Max: -1912765.951ms; Mean: -1912765.951ms; StdDev: -2147483.648ms


Round Trip Jitter
~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 9
    Min: 0.000ms; Max: 0.001ms; Mean: 0.000ms; StdDev: 0.000ms


One-way Jitter (Source->Dest)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 9
    Min: 0.000ms; Max: 0.000ms; Mean: 0.000ms; StdDev: 0.000ms


One-way Jitter (Dest->Source)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 9
    Min: 0.000ms; Max: 0.001ms; Mean: 0.000ms; StdDev: 0.000ms


RP/0/RP0/CPU0:ios#ethernet sla on-demand operation type cfm-syn probe domain DOWN0 source
interface tenGigE 0/0/0/10.1 target mep-id 6001
```

```
Mon Sep 11 12:12:39.259 UTC
Warning: Burst configuration is present and so this profile cannot be represented in the
MEF-SOAM-PM-MIB configuration tables. However, the statistics are still collected
On-demand operation 2 succesfully created
 / - Completed - statistics will be displayed shortly.
RP/0/RP0/CPU0:ios#show ethernet sla statistics on-demand id 2

Mon Sep 11 12:13:24.825 UTC
Source: Interface TenGigE0/0/0/10.1, Domain DOWN0
Destination: Target MEP-ID 6001
================================================================================
On-demand operation ID #2, packet type 'cfm-synthetic-loss-measurement'
Started at 12:12:41 UTC Mon 11 September 2017, runs once for 10s
Frame Loss Ratio calculated every 10s

One-way Frame Loss (Source->Dest)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:12:41 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 100; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                   Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 1
    Min: 0.000%; Max: 0.000%; Mean; 0.000%; StdDev: 0.000%; Overall: 0.000%


One-way Frame Loss (Dest->Source)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:12:41 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 100; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                   Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 1
    Min: 0.000%; Max: 0.000%; Mean; 0.000%; StdDev: 0.000%; Overall: 0.000%

RP/0/RP0/CPU0:ios#show ethernet cfm local meps verbose
Mon Sep 11 12:13:04.461 UTC
Domain UP6 (level 6), Service s6
Up MEP on FortyGigE0/0/1/2.1 MEP-ID 1
================================================================================
  Interface state: Up      MAC address: 008a.960f.c4a8
  Peer MEPs: 1 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 0 missing, 0 unexpected

  CCM generation enabled:  Yes, 1s (Remote Defect detected: No)
                           CCM processing offloaded to hardware
  AIS generation enabled:  No
  Sending AIS:             No
  Receiving AIS:           No

  No packets sent/received

Domain DOWN0 (level 0), Service s10
Down MEP on TenGigE0/0/0/10.1 MEP-ID 2001
================================================================================
  Interface state: Up      MAC address: 008a.960f.c428
  Peer MEPs: 1 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 0 missing, 0 unexpected

  CCM generation enabled:  Yes, 1s (Remote Defect detected: No)
                           CCM processing offloaded to hardware
  AIS generation enabled:  No
  Sending AIS:             No
```

```
      Receiving AIS:            No

      Packet        Sent      Received
      ------   ----------   --------------------------------------------------------
      DMM            10              0
      DMR             0             10
      SLM           100              0
      SLR             0            100
```

# Synthetic Loss Measurement

The synthetic loss measurement mechanism defined in Y.1731 can only be used in point-to-point networks, and only works when there is sufficient flow of data traffic. The difficulties with the Y.1731 loss measurement mechanism was recognized across the industry and hence an alternative mechanism has been defined and standardized for measuring loss of traffic.

This alternative mechanism does not measure the loss of the actual data traffic, but instead injects synthetic CFM frames and measures the loss of these synthetic frames. You can perform a statistical analysis to give an approximation of the loss of data traffic. This technique is called Synthetic Loss Measurement (SLM). SLM has been included in the latest version of the Y.1731 standard. Use SLA to perform the following measurements:

- One-way loss (Source to Destination)

- One-way loss (Destination to Source)

SLM supports the following:

- All Layer 2 transport interfaces, such as physical, bundle interfaces, Layer2 sub-interfaces, pseudowire Head-end interfaces or attachment circuits. Transport network can be EVPN or BGP-MPLS.

- Up and Down MEPs.

- Transparent passing of the SLM packets through the MIP without punting it to the software.

## Configuring Synthetic Loss Measurement

The following section describes how you can configure Synthetic Loss Measurement:

RP/0/RP0/CPU0:router **(config)# ethernet sla**

```
profile test-slm type cfm-synthetic-loss-measurement
 probe
  send packet every 1 seconds
  synthetic loss calculation packets 24
 !
 schedule
  every 3 minutes for 120 seconds
 !
 statistics
  measure one-way-loss-sd
   buckets size 1 probes
   buckets archive 5
  !
  measure one-way-loss-ds
   buckets size 1 probes
   buckets archive 5
!
!
```

```
!
!
interface TenGigE0/0/0/10.1 l2transport
encapsulation dot1q 1
ethernet cfm
  mep domain DOWN0 service s10 mep-id 2001
   sla operation profile test-slm target mep-id 6001
  !
```

### Configuring an On-Demand Ethernet SLA Operation for CFM Synthetic Loss Measurement

To configure an on-demand Ethernet SLA operation for CFM synthetic loss measurement, use this command in privileged EXEC configuration mode:

RP/0/RP0/CPU0:router (config) # **ethernet sla on-demand operation type cfm-synthetic-loss-measurement probe domain D1 source interface TenGigE 0/0/0/0 target mac-address 2.3.4**

### Running Configuration

```
RP/0/RP0/CPU0:router# show ethernet sla statistics on-demand id 1
Mon Sep 11 12:12:00.699 UTC
Source: Interface TenGigE0/0/0/10.1, Domain DOWN0
Destination: Target MEP-ID 6001
================================================================================
On-demand operation ID #1, packet type 'cfm-delay-measurement'
RP/0/RP0/CPU0:router#
RP/0/RP0/CPU0:router# show running-config
Mon Sep 11 12:10:18.467 UTC
Building configuration...
!! IOS XR Configuration version = 6.4.1.14I
!! Last configuration change at Mon Sep 11 12:08:16 2017 by root
!
logging console disable
telnet vrf default ipv4 server max-servers 10
username root
group root-lr
group cisco-support
secret 5 $1$QJT3$94M5/wK5J0v/lpAu/wz31/
!
line console
exec-timeout 0 0
!
ethernet cfm
domain UP6 level 6 id null
  service s6 xconnect group g1 p2p p1 id number 6
   mip auto-create all ccm-learning
   continuity-check interval 1s
   mep crosscheck
    mep-id 4001
   !
  !
!
domain DOWN0 level 0 id null
 service s10 down-meps id number 10
   continuity-check interval 1s
   mep crosscheck
    mep-id 6001
   !
  !
!
```

```
 !
profile test-slm type cfm-synthetic-loss-measurement
  probe
   send packet every 1 seconds
   synthetic loss calculation packets 24
  !
  schedule
   every 3 minutes for 120 seconds
  !
  statistics
   measure one-way-loss-sd
    buckets size 1 probes
    buckets archive 5
   !
   measure one-way-loss-ds
    buckets size 1 probes
    buckets archive 5
!

interface TenGigE0/0/0/10.1 l2transport
encapsulation dot1q 1
ethernet cfm
  mep domain DOWN0 service s10 mep-id 2001
   sla operation profile DMM target mep-id 6001
   sla operation profile test-slm target mep-id 6001
  !
!
!

interface FortyGigE0/0/1/2.1 l2transport
 encapsulation dot1q 1
 ethernet cfm
  mep domain UP6 service s6 mep-id 1
   sla operation profile DMM target mep-id 6001
   sla operation profile test-slm target mep-id 6001
  !
 !
!
l2vpn
 xconnect group g1
  p2p p1
   interface TenGigE0/0/0/10.1
   interface FortyGigE0/0/1/2.1
  !
 !
!
end
```

## Verification

```
Round Trip Delay
~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 10
    Min: 0.009ms; Max: 0.010ms; Mean: 0.009ms; StdDev: 0.000ms


One-way Delay (Source->Dest)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket
```

```
Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                  Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 10
    Min: 1912765.961ms; Max: 1912765.961ms; Mean: 1912765.961ms; StdDev: -2147483.648ms


One-way Delay (Dest->Source)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                  Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 10
    Min: -1912765.952ms; Max: -1912765.951ms; Mean: -1912765.951ms; StdDev: -2147483.648ms


Round Trip Jitter
~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                  Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 9
    Min: 0.000ms; Max: 0.001ms; Mean: 0.000ms; StdDev: 0.000ms


One-way Jitter (Source->Dest)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                  Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 9
    Min: 0.000ms; Max: 0.000ms; Mean: 0.000ms; StdDev: 0.000ms


One-way Jitter (Dest->Source)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 probes per bucket

Bucket started at 12:11:19 UTC Mon 11 September 2017 lasting 10s
    Pkts sent: 10; Lost: 0 (0.0%); Corrupt: 0 (0.0%);
                  Misordered: 0 (0.0%); Duplicates: 0 (0.0%)
    Result count: 9
    Min: 0.000ms; Max: 0.001ms; Mean: 0.000ms; StdDev: 0.000ms
```