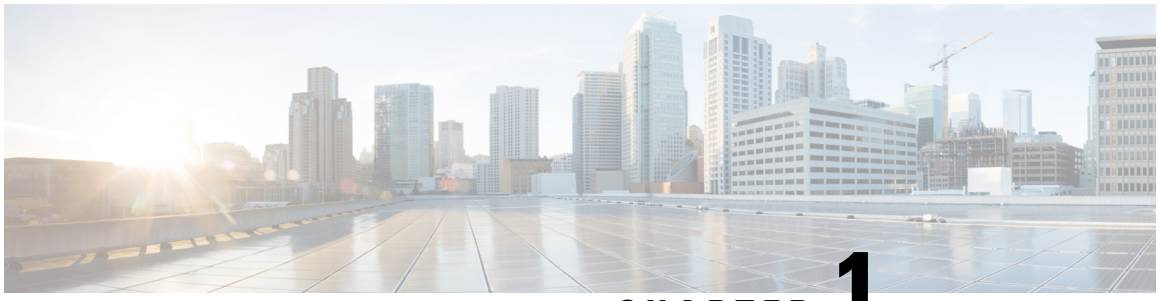
Layer 2 Configuration Guide, Cisco IOS XE Everest 3.18SP (Cisco NCS 4200 Series)

First Published: 2016-07-29

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

CONTENTS



CHAPTER 1

Configuring Ethernet Dataplane Loopback

Ethernet data plane loopback provides a means for remotely testing the throughput of an Ethernet port.

Prerequisites for Ethernet Data Plane Loopback

- Ethernet loopback sessions are supported only of EFPs (service instances, Ethernet flow points, EVCs).
- Dot1q tags must be configured while configuring Ethernet loopback sessions on EFPs. However, loopback sessions can be configured using dot1q/QinQ, even if the underlying EFP has the dot1q/QinQ range configured.
- Internal loopback sessions configured must be within the 1 GB reserved bandwidth.
- Internal loopback can be launched even when the physical interface port state is down.

Restrictions for Ethernet Data Plane Loopback

- Data plane loopback on routed port infrastructure is *not* supported.
- Etype, src-mac, and llc-oui based loopback traffic filtering is *not* supported.
- Port-level QoS is *not* bypassed.
- Port shaper cannot be bypassed in facility loopback.
- Facility and terminal Ethernet data plane loopback (ELB) are *not* supported on dot1ad nni interface.
- Internal loopback sessions configured must be within the 1 GB reserved bandwidth for Cisco ASR 900 Series RSP2 Module.

- A maximum number of 20 facility loopback sessions can be created per system, provided 16 sessions are with Dot1Q and 4 sessions are with Dot1Q and destination MAC address. This scale reduces if SPAN or RSPAN is configured. This scale is supported on the Cisco ASR 900 Series RSP2 module.
- A maximum number of 12 terminal loopback sessions can be created per system, provided 8 sessions are with Dot1Q and 4 sessions are with Dot1Q and destination MAC address. This scale reduces if RSPAN or SADT is configured. This scale is supported on the Cisco ASR 900 Series RSP2 module.
- Only one Ethernet loopback (terminal or facility) session can be active on an EFP at any instance.
- Local SPAN and ELB cannot be enabled on a physical interface at the same time.
- Loopback sessions cannot be initiated on a port configured with SPAN or RSPAN.
- Ethernet loopback is not supported on a range of dot1q tags.
- Ethernet Data Plane Loopback is affected on STP enabled interface.
- Dynamic addition of rewrite ingress tags with default EFP is not supported.
- Dynamic changes at EFP and interface level are not supported when Ethernet Data Plane Loopback is active.
- Egress EFP is not updated for external Ethernet data plane loopback statistics.
- For internal Ethernet data plane loopback ingress and egress interface statistics are not updated on interface, where internal ELB is enabled.

RSP3 Module

- Starting from Cisco IOS XE Amsterdam 17.1.x release, the template sr_5_label_push_enable is not supported with Ethernet loopback.
- Etype, VLAN, COS, src-mac, and llc-oui based loopback traffic filtering is *not* supported.
- Port-based ELB is *not* supported.
- Internal ELB is *not* supported when the physical interface port state is down.
- Data filtering of loopback is *not* enforced for the traffic coming in the opposite direction.
- Filtering based on specific VLAN is *not* supported.
- Dot1Q filter is *not* supported.
- Internal loopback sessions configured must be within the 100 GB reserved recycle bandwidth.
- MAC-ACL *cannot* be bypassed in with facility loopback.
- A maximum number of 20 facility loopback and 12 terminal loopback sessions are supported.

Information on Ethernet Data Plane Loopback

The Ethernet data plane loopback feature provides a means for remotely testing the throughput of an Ethernet port. You can verify the maximum rate of frame transmission with no frame loss. This feature allows for

bidirectional or unidirectional throughput measurement, and on-demand/out-of-service (intrusive) operation during service turn-up. This feature supports two types of Ethernet loopback.

- Facility loopback (external)—Traffic loopback occurs at the Ingress interface. Traffic does not flow into the router for loopback.
- Terminal loopback (internal)—Traffic loopback occurs at the Egress interface. Traffic loopback occurs after the traffic flows into the router to the other interface.

QoS Support for Ethernet Data Plane Loopback

- Ingress QoS is bypassed in external loopback on service instances.
- Internal loopback sequence is as follows:
  - Ingress QoS
  - Egress QoS (egress port) (both, shaper and policer are supported).
- All port-level and EFP-level QoS is applicable for internal Ethernet data plane loopback.
- For external Ethernet data plane loopback:
  - All port-level and EFP-level QoS is bypassed except for shaper.
  - Port-level shaper cannot be bypassed.

How to Configure Ethernet Data Plane Loopback

Enabling Ethernet Data Plane Loopback

```
enable
configure terminal
interface gigabitethernet 0/2/1
service instance 1 ethernet
encapsulation dot1q 100
bridge-domain 120
ethernet loopback permit external
end
```

Note ELB is supported using a MAC filter for UP-MEP session. If you are starting ELB without the MAC filter, the UP-MEP session will go DOWN.

Starting an Ethernet Data Plane Loopback Session

Note To start a loopback for untagged and default EFPs, dot1q and second-dot1q are not needed.

Note By default the session would be running for 300 seconds unless you explicitly specify and automatically stops after the session time expiry.

```
enable
configure terminal
ethernet loopback start local interface gigabitEthernet 0/4/1 service instance 10 external
 dot1q 10 cos 1 destination mac-address 0000.0000.0001 timeout none
end
This is an intrusive loopback and the packets matched with the service will not be able
to pass through.
Continue? (yes/[no]): yes
```

Stopping an Active Session

Use the **ethernet loopback stop** command to stop an active session on an interface or to stop all sessions based on the session id.

```
Router# ethernet loopback stop local interface gigabitEthernet 0/4/1 id 1
```

Configuration Examples

Example: Configuring External Loopback

This example shows how to configure external (facility) loopback.

```
Router(config)# interface gigabitEthernet 0/4/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 120
Router(config-if-srv)# bridge-domain 120
Router(config-if-srv)# ethernet loopback permit external
```

This example shows external (facility) loopback on the Gigabit Ethernet 0/4/1 interface:

```
interface GigabitEthernet0/4/1
 no ip address
 negotiation auto
 service instance 10 ethernet
  encapsulation dot1q 10
  rewrite ingress tag pop 1 symmetric
  bridge-domain 10
  ethernet loopback permit external ===? For facility loopback
 !
end
```

This example below shows how to start external (facility) loopback on the router. A warning message is displayed. Type **yes** to continue.

```
Router# ethernet loopback start local interface gigabitEthernet 0/4/1 service instance 10
external dot1q 10 cos 1
 destination mac-address 0000.0000.0001 timeout none

This is an intrusive loopback and the packets matched with the service will not  be able
```

```
to pass through.
Continue? (yes/[no]): yes
```

Example: Configuring Terminal Loopback

This example shows how to configure internal (terminal) loopback.

```
Router(config)# interface gigabitEthernet 0/0/0
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 120
Router(config-if-srv)# bridge-domain 120
Router(config-if-srv)# ethernet loopback permit internal
```

This example shows internal (terminal) loopback on Gigabit Ethernet 0/0/0 interface:

```
interface TenGigabitEthernet0/0/0
 no ip address
service instance 10 ethernet
  encapsulation dot1q 10
  rewrite ingress tag pop 1 symmetric
  bridge-domain 10
  ethernet loopback permit internal
 !
end
```

Verifying Ethernet Data Plane Loopback

Example: Verifying Ethernet Dataplane Loopback

Use the **show ethernet loopback** {**active** | **permitted**} [**interface** *interface numbe*r] command.

- The following example displays the loopback capabilities per interface. The output shows internal (terminal) loopback has been permitted on Ten Gigabit Ethernet 0/0/0 interface and external (facility) loopback has been permitted on Gigabit Ethernet 0/4/1 interface.

```
Router# show ethernet loopback permitted

-------------------------------------------------------------------------------
Interface                               SrvcInst Direction
Dot1q/Dot1ad(s)                         Second-Dot1q(s)
-------------------------------------------------------------------------------
Te0/0/0                                 10                              Internal
10
Gi0/4/1                                 10                              External
10
```

- This example shows all active sessions on the router.

```
Router# show ethernet loopback active

============================================================
Loopback Session ID        : 1
Interface                  : GigabitEthernet0/4/1
Service Instance           :10
Direction                  : External
Time out(sec)              : none
Status                     : on
Start time                 : 10:31:09.539 IST Mon Aug 26 2013
Time left                  : N/A
```

```
Dot1q/Dot1ad(s)            : 10
Second-dot1q(s)            :
Source Mac Address         : Any
Destination Mac Address    : 0000.0000.0001
Ether Type                 : Any
Class of service           : 1
Llc-oui                    : Any


Total Active Session(s)    : 1
Total Internal Session(s)  : 0
Total External Session(s)  : 1
```

- This example shows how to stop the sessions on the router.

```
Router# ethernet loopback stop local interface GigabitEthernet
0/4/1 id 1
```

Use Cases or Deployment Scenarios

ELB is Supported with MAC Filter for UP-MEP Session

In the following scenario, ELB is supported using a MAC filter for UP-MEP session. If you starting ELB with out MAC filter, the UP-MEP session will go DOWN.

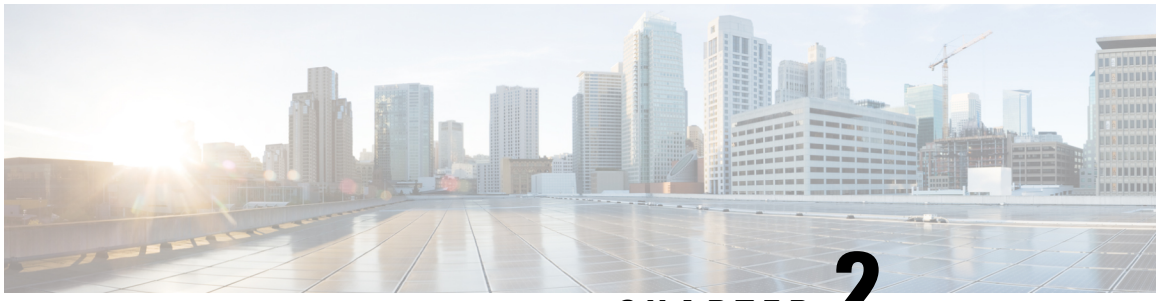
```
enable
configure terminal
service instance 800 ethernet 800
encapsulation dot1q 800
service-policy input <NAME>
xconnect 2.2.2.2 880 encapsulation mpls
cfm mep domain <NAME> mpid 200
cos 7
ethernet loopback permit external
ethernet loopback permit internal

Router#ethernet loopback start local interface gi0/0/0 service instance 800 internal dot1q
 800 destination mac-address f078.1685.313f timeout none

This is an intrusive loopback and the packets matched with the service will not  be able
to pass through. Continue? (yes/[no]): yes


Router#show ethernet cfm maintenance-points remote
-------------------------------------------------------------------------------
MPID  Domain Name                                MacAddress          IfSt  PtSt
 Lvl  Domain ID                                  Ingress
 RDI  MA Name                                    Type Id             SrvcInst
      EVC Name                                                       Age
      Local MEP Info
-------------------------------------------------------------------------------
220   CCI                                        f078.1685.313f      Up    Up
 0    CCI                                        Gi0/0/0:(2.2.2.2, 880)
 -    800                                        XCON N/A            800
      800                                                            0s
      MPID: 200 Domain: CCI MA: 800

Total Remote MEPs: 1
```



CHAPTER 2

Configuring Switched Port Analyzer

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

Local SPAN Session

A local Switched Port Analyzer (SPAN) session is an association of a destination interface with a set of source interfaces. You configure local SPAN sessions using parameters that specify the type of network traffic to monitor. Local SPAN sessions allow you to monitor traffic on one or more interfaces and to send either ingress traffic, egress traffic, or both to one destination interface.

Local SPAN sessions do not interfere with the normal operation of the switch. You can enable or disable SPAN sessions with command-line interface (CLI) commands. When enabled, a local SPAN session might become active or inactive based on various events or actions, and this would be indicated by a syslog message. The **show monitor session span** *session number* command displays the operational status of a SPAN session.

A local SPAN session remains inactive after system power-up until the destination interface is operational.

The following configuration guidelines apply when configuring local SPAN:

- When enabled, local SPAN uses any previously entered configuration.
- Use the **no monitor session** *session number* command with no other parameters to clear the local SPAN session number.

RSPAN Session

An RSPAN source session is an association of source ports or VLAN across your network with an RSPAN Vlan. The RSPAN VLAN/BD on the router is the destination RSPAN session.

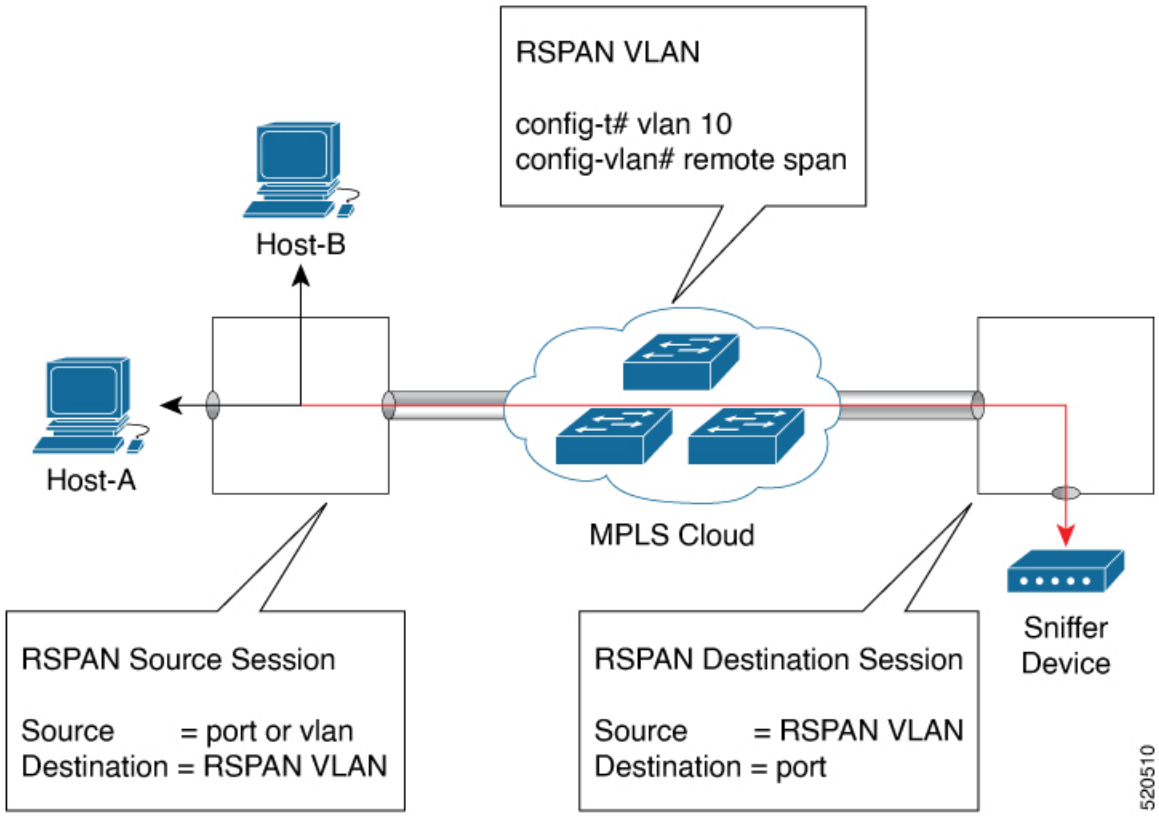
RSPAN over VPLS Network for RSP3 Module

Table 1: Feature History

Feature Name	Release	Description
RSPAN over VPLS Pseudowire Network	Cisco IOS XE Amsterdam 17.3.1	This feature allows the traffic mirroring destination port to be configured as a pseudowire rather than a physical port. This feature lets the designated traffic on the source port to be mirrored over the pseudowire to a remote location. This feature is supported on the Cisco RSP3 module.

RSPAN allows remote traffic monitoring, where the source and destination routers are connected by VPLS pseudowire network. The SPAN Source and Destination routers are connected through a VPLS Pseudowire connected with the RSPAN VLAN over an MPLS or IP network. The VPLS pseudowire is dedicated only to the RSPAN traffic. All the mirrored traffic from the source port is carried over the VPLS Pseudowire connected with the RSPAN VLAN towards the destination port. On the destination router, a port belonging to the RSPAN VLAN or EVC BD is connected to the sniffer device.

Figure 1: RSPAN Traffic over VPLS Network on the Cisco RSP3 module



Prerequisites for Configuring Local Span and RSPAN

Local Span

- Use a network analyzer to monitor interfaces.

RSPAN

- Before configuring RSPAN sessions, you must first configure:
  1. Source interface
  2. Destination BD
- MAC learning should be disabled using the mac-address-table limit *[rspan vlan/bd]* maximum num action limit command before configuring the RSPAN VLAN.
- RSPAN VLAN must be dedicated and entire Layer 2 devices in the network must be aware of the VLAN.
- RSPAN source and destinations switches separated by the VPLS pseudowire must be aware of the RSPAN VLAN/ brige domain (BD).
- Pseudowire must be dedicated for RSPAN traffic.

- The RSPAN destination session is not required when the destination switch is connected to source switch through Layer2 VPN. Thus, in the destination switch, the destination port must configured with the service instance with encapsulation as RSPAN VLAN/BD and bridge domain as RSPAN VLAN/BD and the MAC address learning should be disabled on RSPAN BD/VLAN.

Restrictions for Local Span and RSPAN

Local Span

- Local SPAN is only supported on physical ports.
- SPAN monitoring of port-channel interfaces or port-channel member-links is *not* supported.
- Combined Egress local SPAN bandwidth supported on Cisco ASR 900 Series RSP2 module is 1 GB.
- Local SPAN is not supported on logical interfaces such as Vlans or EFPs.
- Up to 14 active local SPAN sessions (ingress and egress) are supported. The router supports up to 14 ingress sessions and up to 12 egress sessions.
- Only one local SPAN destination interface is supported. You *cannot* configure a local SPAN destination interface to receive ingress traffic.
- Outgoing Cisco Discovery Protocol (CDP) and Bridge Protocol Data Unit (BPDU) packets are not replicated.
- When enabled, local SPAN uses any previously entered configuration.
- When you specify source interfaces and do not specify a traffic direction (**Tx**, **Rx**, or **both**), **both** is used by default.
- The SPAN port does not work for Rx traffic on the pseudowire for interfaces, when the SPAN port is in different ASIC of the RSP2 module.
- Local SPAN destinations never participate in any spanning tree instance. Local SPAN includes BPDUs in the monitored traffic, so any BPDUs seen on the local SPAN destination are from the local SPAN source.
- Local SPAN sessions with overlapping sets of local SPAN source interfaces or VLANs are *not* supported.

RSP3 module

- Destination port of SPAN session, *cannot* be used for other network data traffic flow.
- Multiple destinations for same SPAN session is *not* supported on the Cisco ASR 900 Series RSP3 module.
- Jumbo sized packets and bad CRC packets are *not* spanned.
- Combined Egress local SPAN bandwidth supported is about 100GB depending on other traffic on the internal recycle interface.
- Port-channel *cannot* be used as the SPAN destination.

RSPAN

- RSPAN Vlan/BD is *not* used for data traffic.
- The maximum number of supported RSPAN sessions are 14.
- Only one source port is supported per RSPAN.
- Source ranges (vlan range or port range) is *not* supported.

- Vlan filtering is not supported.
- If two RSPAN configurations sessions are configured on two RSPAN BDs associated to the same Trunk EFP, the traffic from the first session flows to the second session after it is configured.
- RSPAN destination configuration for Layer2 pseudowire is *not* supported.
- If RSPAN BD is associated with a VPLS pseudowire, the traffic flows through the VPLS pseudowire.
- If RSPAN source and destination are separated by pseudowire, then the RSPAN VLAN details must be updated to both RSPAN source switch and destination switch. The pseudowire should also be dedicated for RSPAN traffic.
- BDI should not be created when that BD is part of RSPAN.
- Monitor session should be created only after RSPAN BD is created.
- Do not have RSPAN bridge domain as part of RSPAN source interface.

RSP3 module

- RSPAN is *not* supported on the Cisco ASR 900 Series RSP3 module.

Understanding Local SPAN and RSPAN

Local SPAN Traffic

Network traffic, including multicast, can be monitored using SPAN. Multicast packet monitoring is enabled by default. In some SPAN configurations, multiple copies of the same source packet are sent to the SPAN destination interface. For example, a bidirectional (both ingress and egress) SPAN session is configured for sources a1 and a2 to a destination interface d1. If a packet enters the switch through a1 and gets switched to a2, both incoming and outgoing packets are sent to destination interface d1; both packets would be the same (unless a Layer-3 rewrite had occurred, in which case the packets would be different).

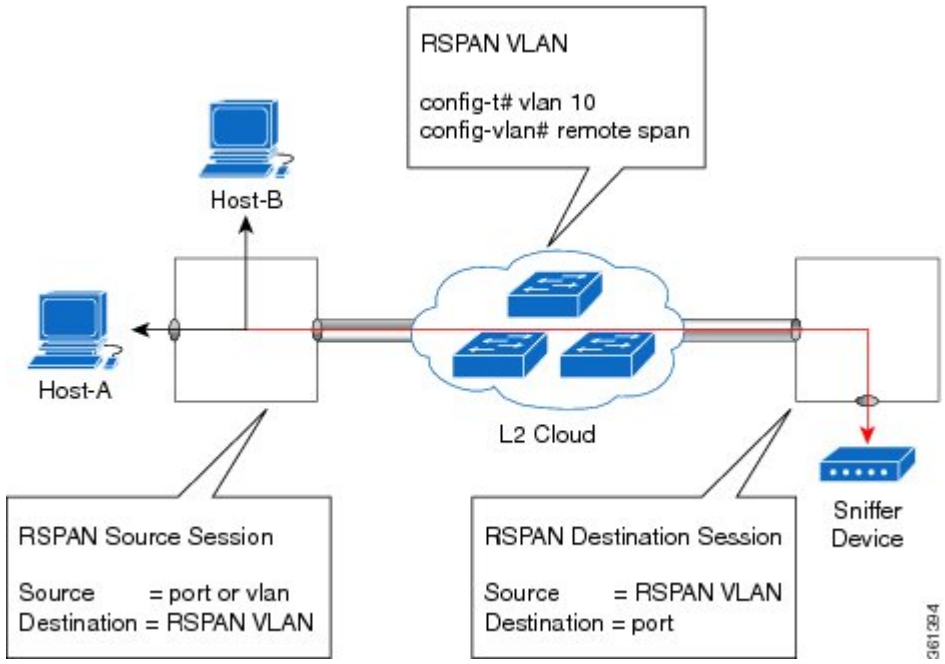
RSPAN Traffic for RSP2 Module

RSPAN supports source ports and source VLANs in the source switch and destination as RSPAN VLAN/BD.

The figure below shows the original traffic from the Host A to Host B via the source ports or VLANs on Host A. The source ports or VLANs of Host A is mirrored to Host B using RSPAN VLAN 10. The traffic for each RSPAN session is carried over a user-specified RSPAN VLAN that is dedicated for that RSPAN session in all participating devices. The traffic from the source ports or VLANs are mirrored into the RSPAN VLAN and forwarded over Trunk or the EVC bridge domain (BD) ports carrying the RSPAN VLAN to a destination session monitoring the RSPAN VLAN.

Each RSPAN source must have either ports or VLANs as RSPAN sources. On RSPAN destination, the RSPAN VLAN is monitored and mirrored to the destination physical port connected to the sniffer device.

Figure 2: RSPAN Traffic



RSPAN allows remote monitoring of traffic where the source and destination switches are connected by L2VPN networks

The RSPAN source is either ports or VLANs as in a traditional RSPAN. However, the SPAN source and destination devices are connected through a L2 pseudowire associated with the RSPAN VLAN over an MPLS/IP network. The L2 pseudowire is dedicated for only RSPAN traffic. The mirrored traffic from the source port or VLAN is carried over the pseudowire associated with the RSPAN VLAN towards the destination side. On the destination side, a port belonging to the RSPAN VLAN or EVC BD is connected to sniffer device.

Destination Interface

A destination interface, also called a monitor interface, is a switched interface to which SPAN or RSPAN sends packets for analysis. You can have only one destination interface for SPAN sessions.

An interface configured as a destination interface cannot be configured as a source interface. Specifying a trunk interface as a SPAN or RSPAN destination interface stops trunking on the interface.

Source Interface

A source interface is an interface monitored for network traffic analysis. An interface configured as a destination interface cannot be configured as a source interface.

Traffic Directions

Ingress SPAN (Rx) copies network traffic received by the source interfaces for analysis at the destination interface. Egress SPAN (Tx) copies network traffic transmitted from the source interfaces to the destination

interface. Specifying the configuration option (both) copies network traffic received and transmitted by the source interfaces to the destination interface.

The following table lists the supported traffic types for RSPAN.

Table 2: RSPAN Traffic

Source	Ingress Mirror (Rx)	Egress Mirror (Tx)	Both
Layer2 or Layer3	Supported	Supported	Supported
VLAN	Supported	Not supported	Not supported
EFP	Not supported	Not supported	Not supported
Pseudowire	Not supported	Not supported	Not supported

The following table lists the supported **rewrite** traffic for RSPAN on the EFP, Trunk with the associated RSPAN bridge domains.

Table 3: Rewrite Traffic for RSPAN BD

Rewrite Operations	Source	EFP/Trunk associated with RSPAN BD
no-rewrite	Pop1, Pop2, Push1	Only Pop1

The following tables lists the format of the spanned packets at the destination port for both Ingress and Egress RSPAN. The tables lists the formats of untagged, single, and double tagged source packets for EFPs under source port configured with **rewrite** operations (no-rewrite, pop1, pop2 and push1).

Table 4: Destination Port Ingress and Egress Spanned Traffic for EVC RSPAN BD

<table>
<tr><th></th><th>Ingress Traffic</th><th>Egress Traffic</th></tr>
<tr><td>(Untagged Traffic) - Source port rewrite</td><td>RSPAN Vlan (BD) rewrite pop1 tag symmetric</td><td>RSPAN Vlan (BD) rewrite pop1 tag symmetric</td></tr>
<tr><td>no-rewrite</td><td>RSPAN BD tag + packet</td><td>RSPAN BD tag + packet</td></tr>
<tr><td>pop1 tag</td><td>NA</td><td>NA</td></tr>
<tr><td>pop2 tag</td><td>NA</td><td>NA</td></tr>
<tr><td>push1 tag</td><td>NA</td><td>NA</td></tr>
<tr><td>(Single Traffic)-Source port rewrite</td><td>RSPAN Vlan (BD) rewrite pop1 tag symmetric</td><td>RSPAN Vlan (BD) rewrite pop1 tag symmetric</td></tr>
<tr><td>no-rewrite</td><td rowspan="4">RSPAN BD tag + source-outer-tag + packet</td><td rowspan="2">RSPAN BD tag + source-outer-tag + packet</td></tr>
<tr><td>pop1 tag</td></tr>
<tr><td>pop2 tag</td><td>NA</td></tr>
<tr><td>push1 tag</td><td>RSPAN BD tag + source-outer-tag + packet</td></tr>
</table>

	Ingress Traffic	Egress Traffic
(Double traffic) - Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + source-outer-tag + source-inner-tag + packet	RSPAN BD tag + Source-inner-tag + packet
pop1 tag		
pop2 tag		
push1 tag		

Table 5: Destination Port Ingress and Egress Spanned Traffic for TEFP RSPAN BD

	Ingress Traffic	Egress Traffic
(Untagged traffic)- Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + packet	RSPAN BD tag + packet
pop1 tag	NA	NA
pop2 tag	NA	NA
push1 tag	NA	NA
(Single traffic)-Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + source-outertag + packet	RSPAN BD tag + source-outertag + packet
pop1 tag		
pop2 tag		NA
push1 tag		RSPAN BD tag + source-outertag + packet
(Double traffic) -Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + source-outertag + source-innertag+ packet	RSPAN BD tag + source-outertag + source-innertag + packet
pop1 tag		
pop2 tag		
push1 tag		

Table 6: Destination Port Ingress and Egress Spanned Traffic for RSPAN BD with VPLS Pseudowire

	Ingress Traffic	Egress Traffic
(Untagged traffic) - Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + packet	RSPAN BD tag + packet
pop1 tag	NA	NA
pop2 tag	NA	NA
push1 tag	NA	NA
(Single traffic)- Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + source-outer-tag + packet	RSPAN BD tag + source-outer-tag + packet
pop1 tag		
pop2 tag	NA	NA
push1 tag	RSPAN BD tag + source-outer-tag + packet	RSPAN BD tag + source-outer-tag + packet
(Double traffic)-Source port rewrite	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**	**RSPAN Vlan (BD) rewrite pop1 tag symmetric**
no-rewrite	RSPAN BD tag + source-outer-tag + source-inner-tag + packet	RSPAN BD tag + source-outer-tag + source-inner-tag + packet
pop1 tag		
pop2 tag		
push1 tag		

Configuring Local SPAN and RSPAN

Configuring Sources and Destinations for Local SPAN

To configure sources and destinations for a SPAN session:

SUMMARY STEPS

1. **configure terminal**
2. **monitor session {***session_number***} type local**
3. **source interface** *interface_type slot/subslot/port* **[, | - | rx | tx | both]**
4. **destination interface** *interface_type slot/subslot/port* **[, | -]**
5. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 2	**monitor session {***session_number***} type local** **Example:** `Router(config)# monitor session 1 type local`	Specifies the local SPAN session number and enters the local monitoring configuration mode. • *session_number*—Indicates the monitor session. The valid range is 1 through 14.
Step 3	**source interface** *interface_type slot/subslot/port* **[, | - | rx | tx | both]** **Example:** `Router(config-mon-local)# source interface gigabitethernet 0/2/1 rx`	Specifies the source interface and the traffic direction: • *interface_type*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface. • *slot/subslot/port*—The location of the interface. • ","—List of interfaces • "–"—Range of interfaces • rx—Ingress local SPAN • tx—Egress local SPAN • both
Step 4	**destination interface** *interface_type slot/subslot/port* **[, | -]** **Example:** `Router(config-mon-local)# destination interface gigabitethernet 0/2/4`	Specifies the destination interface that sends both ingress and egress local spanned traffic from source port to the prober or sniffer. • *interface_type*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface. • *slot/subslot/port*—The location of the interface. • ","—List of interfaces • "–"—Range of interfaces
Step 5	**no shutdown** **Example:** `Router(config-mon-local)# no shutdown`	Enables the local SPAN session.

Removing Sources or Destinations from a Local SPAN Session

To remove sources or destinations from a local SPAN session, use the following commands beginning in EXEC mode:

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **no monitor session** *session-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** *session-number* **Example:** `Router(config)# no monitor session 2`	Clears existing SPAN configuration for a session.

Configuring RSPAN Source Session

To configure the source for a RSPAN session:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *RSPAN_source_session_number* **type rspan-source**
4. **source** {*single_interface* slot/subslot/port| *single_vlan* [**rx** | **tx** | **both**]
5. **destination remote vlan** *rspan_vlan_ID*
6. **no shutdown**
7. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**monitor session** *RSPAN_source_session_number* **type rspan-source** **Example:** `Router(config)# monitor session 1 type rspan-source`	Configures an RSPAN source session number and enters RSPAN source session configuration mode for the session. • *RSPAN_source_session_number*—Valid sessions are 1 to 14. • **rspan-source**—Enters the RSPAN source-session configuration mode.
Step 4	**source** {*single_interface* slot/subslot/port| *single_vlan* [**rx** | **tx** | **both**] **Example:** `Router(config-mon-rspan-src)# source interface gigabitethernet 0/2/1 tx`	Specifies the RSPAN session number, the source interfaces and the traffic direction to be monitored. • *single_interface*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface. • *slot/subslot/port*—The location of the interface. • *single_vlan*—Specifies the single VLAN. • **both**—(Optional) Monitors the received and the transmitted traffic. • **rx**—(Optional) Monitors the received traffic only. • **tx**—(Optional) Monitors the transmitted traffic only.
Step 5	**destination remote vlan** *rspan_vlan_ID* **Example:** `Router(config-mon-rspan-src)# destination remote vlan2`	Associates the RSPAN source session number session number with the RSPAN VLAN. • *rspan_vlan_ID*—Specifies the Vlan ID. **Note** *rspan_vlan_ID* is the RSPAN BD that is configured under the EFP or port which carries the RSPANd traffic.
Step 6	**no shutdown** **Example:** `Router(config-mon-rspan-src)# no shutdown`	Restarts the interface.
Step 7	**end** **Example:** `Router(config-mon-rspan-src)# end`	Exists the configuration.

Configuring RSPAN Destination Session

To configure the destination for a RSPAN session for remote Vlan:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *RSPAN_destination_session_number* **type rspan-destination**

4. **source remote vlan** *rspan_vlan_ID*
5. **destination** {*single_interface slot/subslot/port}*
6. **no shutdown**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**monitor session** *RSPAN_destination_session_number* **type rspan-destination** **Example:** `Router(config)# monitor session 1 type rspan-destination`	Configures a RPAN session. • *RSPAN_destination_session_number*—Valid sessions are 1 to 80. • **rspan-destination**—Enters the RSPAN destination-session configuration mode.
Step 4	**source remote vlan** *rspan_vlan_ID* **Example:** `Router(config-mon-rspan-dst)# source remote vlan2`	Associates the RSPAN destination session number RSPAN VLAN. • *rspan_vlan_ID*—Specifies the Vlan ID
Step 5	**destination** {*single_interface slot/subslot/port}* **Example:** `Router(config-mon-rspan-dst)# destination interface gigabitethernet 0/0/1`	Associates the RSPAN destination session number with the destination port. • *single_interface* —Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface. • *slot/subslot/port*—The location of the interface.
Step 6	**no shutdown** **Example:** `Router(config-mon-rspan-dst)# no shutdown`	Restarts the interface
Step 7	**end** **Example:** `Router(config-mon-rspan-dst)# end`	Exists the configuration

Removing Sources or Destinations from a RSPAN Session

To remove sources or destinations from a RSPAN session:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *session_number*
4. **no** {**source** | **destination}** {*single_interface slot/subslot/port* | *single_vlan*} [**,** | **- both** | **rx** | **tx**]
5. **no monitor session** *session number*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**monitor session** *session_number* **Example:** `Router(config)# monitor session 1`	Configures an RSPAN source session number and enters RSPAN source session configuration mode for the session. • *session_number*—The valid sessions are 1 through 14.
Step 4	**no** {**source** | **destination}** {*single_interface slot/subslot/port* | *single_vlan*} [**,** | **- both** | **rx** | **tx**] **Example:** `Router(config-mon-rspan-src)# no source interface gigabitethernet 0/0/1 tx`	Specifies the RSPAN session number, the source interfaces and the traffic direction to be removed. • **source**—Enters source interfaces. • *slot/subslot/port*—The location of the interface. • *single_interface*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface. • *single_vlan*—Specifices the Vlan. • **both**—(Optional) Monitors the received and the transmitted traffic. • **rx**—(Optional) Monitors the received traffic only. • **tx**—(Optional) Monitors the transmitted traffic only.
Step 5	**no monitor session** *session number* **Example:** `Router(config)# no monitor session 1`	Exits monitor session.

	Command or Action	Purpose
Step 6	**end** **Example:** `Router(config-mon-rspan-src)# end`	Exits configuration mode.

Configuring RSPAN Source Session over VPLS Network

To configure the source for a RSPAN over VPLS Network:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *RSPAN_source_session_number* **type rspan-source**
4. **no shutdown**
5. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **monitor session** *RSPAN_source_session_number* **type rspan-source**

Example:

```
Router(config)#source int g0/0/1 [tx |rx|both]
Router(config)#destination remote VLAN 1000
```

Configures an RSPAN source session number and enters RSPAN over VPLS Network source session configuration mode for the session.

Step 4 **no shutdown**

Example:

```
Router(config-mon-rspan-src)# no shutdown
```

Enables RSPAN over VPLS Network source.

Step 5 **end**

Example:

```
Router(config-mon-rspan-src)# end
```

Exits the configuration.

Note You must ensure that the BDI number should match RSPAN destination remote VLAN number.

Configuring L2VPN VFI in RSPAN Source Session

To configure the source for a RSPAN over VPLS Network using L2VPN Virtual Forwarding Instance (VFI):

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context VPLS** *1000*
4. **source interface***[tx |rx|both]*
5. **no shutdown**
6. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Step 3 **l2vpn vfi context VPLS** *1000*

Example:

```
l2vpn vfi context VPLS1000
 vpn id 1000
 member 10.0.0.1 encapsulation mpls
```

```
source int g0/0/1 [tx |rx|both]
```

Step 4 **source interface***[tx |rx|both]*

Example:

```
l2vpn vfi context VPLS1000
 vpn id 1000
 member 10.0.0.1 encapsulation mpls

source int g0/0/1 [tx |rx|both]
```

Step 5 **no shutdown**

Example:

```
Router(config-mon-rspan-sour)# no shutdown
```

Enables RSPAN over VPLS Network at source.

Step 6 **end**

Example:

```
Router(config-mon-rspan-sour)# end
```

Exits the configuration.

Configuring L2VPN VFI in RSPAN Destination Session

To configure the destination for a RSPAN over VPLS Network using L2VPN Virtual Forwarding Instance (VFI):

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context VPLS***1000*
4. **destination interface***[tx |rx|both]*
5. **no shutdown**
6. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Step 3 **l2vpn vfi context VPLS***1000*

Example:

```
l2vpn vfi context VPLS1000
 vpn id 1000
 member 10.0.0.2 encapsulation mpls

bridge-domain 1000
 member GigabitEthernet0/1/0 service-instance 1
 member vfi VPLS1000
```

Step 4 **destination interface***[tx |rx|both]*

Example:

```
l2vpn vfi context VPLS1000
 vpn id 1000
 member 10.0.0.2 encapsulation mpls

destination int g0/0/1 [tx |rx|both]
```

Step 5 **no shutdown**

Example:

```
Router(config-mon-rspan-dest)# no shutdown
```

Enables RSPAN over VPLS Network at destination.

Step 6 **end**

Example:

```
Router(config-mon-rspan-dest)# end
```

Exits the configuration.

Configuring MAC Limit on RSPAN over VPLS Network in RSPAN Destination Session

To configure the MAC Limit 0 on RSPAN BD on destination router:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mac-address-table-limit** *bdomain <bd-id> maximum 0 action limit*

DETAILED STEPS

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Step 3 **mac-address-table-limit** *bdomain <bd-id> maximum 0 action limit*

Example:

```
router#show bridge-domain 1000
Bridge-domain 1000 (2 ports in all)
State: UP
Mac learning: Enabled
Aging-Timer: 300 second(s)

Maximum address limit: 0
    GigabitEthernet0/1/0 service instance 1
    vfi VPLS1000 neighbor 10.0.0.2 1000
```

Sample Configurations

The following sections contain configuration examples for SPAN and RSPAN.

Configuration Example: Local SPAN

The following example shows how to configure local SPAN session 8 to monitor bidirectional traffic from source interface Gigabit Ethernet interface to destination:

```
Router(config)# monitor session 8 type local
Router(config)# source interface gigabitethernet 0/0/10
Router(config)# destination interface gigabitethernet 0/0/3
Router(config)# no shut
```

Configuration Example: Removing Sources or Destinations from a Local SPAN Session

This following example shows how to remove a local SPAN session:

```
Router(config)# no monitor session 8
```

Configuration Example: RSPAN Source

The following example shows how RSPAN session 2 to monitor bidirectional traffic from source interface Gigabit Ethernet 0/0/1:

```
Router(config)# monitor session 2 type RSPAN-source
Router(config-mon-RSPAN-src)# source interface gigabitEthernet0/0/1 [tx |rx|both]
Router(config-mon-RSPAN-src)# destination remote VLAN 100
Router(config-mon-RSPAN-src)# no shutdown
Router(config-mon-RSPAN-src)# end
```

The following example shows how RSPAN session 3 to monitor bidirectional traffic from source Vlan 20:

```
Router(config)# monitor session 3 type RSPAN-source
Router(config-mon-RSPAN-src)# source VLAN 20 rx
Router(config-mon-RSPAN-src)# destination remote VLAN 100
Router(config-mon-RSPAN-src)# no shutdown
Router(config-mon-RSPAN-src)# end
```

Configuration Example: RSPAN Destination

The following example shows how to configure interface Gigabit Ethernet 0/0/1 as the destination for RSPAN session 2:

```
Router(config)# monitor session 2 type RSPAN-destination
Router(config-mon-RSPAN-dst)# source remote VLAN 100
Router(config-mon-RSPAN-dst)# destination interface gigabitEthernet 0/0/1
Router(config-mon-RSPAN-dst)# end
```

Verifying Local SPAN and RSPAN

Use the **show monitor session** command to view the sessions configured.

- The following example shows the Local SPAN source session with Tx as source:

```
Router# show monitor session 8
Session 8
---------
Type : Local Session
Status : Admin Enabled
Source Ports :
TX Only : Gi0/0/10
Destination Ports : Gi0/0/3
MTU : 1464
Dest RSPAN VLAN : 100
```

- The following example shows the RSPAN source session with Gigabit Ethernet interface 0/0/1 as source:

```
Router# show monitor session 2
Session 2
```

```
---------
Type                    : Remote Source Session
Status                  : Admin Enabled
Source Ports            :
    Both                : Gi0/0/1
MTU                     : 1464
```

- The following example shows the RSPAN source session with Vlan 20 as source:

```
Router# show monitor session 3
Session 3
---------
Type                    : Remote Source Session
Status                  : Admin Enabled
Source VLANs            :
    RX Only             : 20
MTU                     : 1464
```

- The following example shows the RSPAN destination session with Gigabit Ethernet interface 0/0/1 as destination:

```
Router# show monitor session 2
Session 2
---------
Type                    : Remote Destination Session
Status                  : Admin Enabled
Destination Ports       : Gi0/0/1
MTU                     : 1464
Source RSPAN VLAN : 100
```

Verifying RSPAN over VPLS Network

Use the **show monitor session** command to view the sessions configured.

The following example shows the RSPAN over VPLS Source session

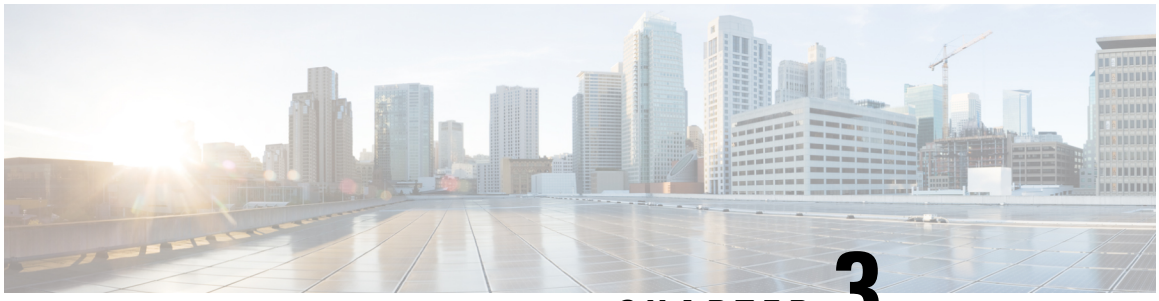
```
Router(config)#show mpls l2transport vc

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
VFI VPLS1000   vfi                        10.0.0.1         1000       UP
```

The following example shows the RSPAN over VPLS Destination session

```
Router(config)#show mpls l2transport vc

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
VFI VPLS1000   vfi                        10.0.0.2         1000       UP
```



CHAPTER 3

Layer 2 Access Control Lists on EVCs

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

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

Prerequisites for Layer 2 Access Control Lists on EVCs

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

Prerequisites for Layer 2 Access Control Lists on EVCs

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

Restrictions for Layer 2 Access Control Lists on EVCs

- A maximum of 512 access control entries (ACEs) are allowed for a given ACL, with the limitation that it does not exceed the maximum tcam entries.
- L2 ACL is supported over port channel with Normal EFPs.
- Egress L2 ACL on EVC is *not* supported.
- L2 ACLs are *not* supported on Trunk EFP.

- L2 ACL counters are *not* supported.
- Layer2 ACL can be applied on layer 2 frame without IPv4 or IPv6 header as layer 2 ACL does not support filter on IPv4 or IPv6 traffic.
- Layer 2 ACLs function inbound only. The Layer 2 ACLs are *not* supported at physical interface level.

EVCs

An Ethernet virtual circuit (EVC) as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint Layer 2 circuit. It is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer. An EVC contains the different parameters on which the service is being offered. A service instance is the instantiation of an EVC on a specified port.

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

Ethernet virtual connection services (EVCS) uses the EVCs and service instances to provide Layer 2 switched Ethernet services. EVC status can be used by a customer edge (CE) device either to find an alternative path to the service provider network or in some cases, to fall back to a backup path over Ethernet or over another alternative service such as ATM.

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

Relationship Between ACLs and Ethernet Infrastructure

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

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

Information About Layer 2 Access Control Lists on EVCs

Creating a Layer 2 ACL

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mac access-list extended** *name*
4. **permit** {{*src-mac mask* | **any**} {*dest-mac mask* | **any**} [*protocol* [**vlan** *vlan*] [*cos value*]]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac access-list extended** *name* **Example:** `Device(config)# mac access-list extended test-12-acl`	Defines an extended MAC ACL and enters mac access list control configuration mode.
Step 4	**permit** {{*src-mac mask* | **any**} {*dest-mac mask* | **any**} [*protocol* [**vlan** *vlan*] [*cos value*]]} **Example:** `Device(config-ext-macl)# permit 00aa.00bb.00cc 0.0.0 any`	Allows forwarding of Layer 2 traffic if the conditions are matched. Creates an ACE for the ACL.

Applying a Layer 2 ACL to a Service Instance

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

Before you begin

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* ethernet

5. **encapsulation dot1q** *vlan-id*
6. **mac access-group** *access-list-name* in
7. **bridge -domain** *bridge-id* in

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/0/0`	Specifies the type and location of the interface to configure, where: • *type* --Specifies the type of the interface. • *number* --Specifies the location of the interface.
Step 4	**service instance** *id* ethernet **Example:** `Device(config-if)# service instance 100 ethernet`	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**mac access-group** *access-list-name* in **Example:** `Device(config-if-srv)# mac access-group test-12-acl in`	Applies a MAC ACL to control incoming traffic on the interface.
Step 7	**bridge -domain** *bridge-id* in **Example:** `Device(config-if-srv)# bridge-domain 100`	Configure the bridge domain ID.

Configuring a Layer 2 ACL with ACEs on a Service Instance

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac access-list extended** *name* **Example:** `Device(config)# mac access list extended test-12-acl`	Defines an extended MAC ACL and enters mac access control list configuration mode.
Step 4	**permit** {*src-mac mask* | **any**} {*dest-mac mask* | **any**} **Example:** `Device(config-ext-macl)# permit 00aa.bbcc.ddea 0.0.0 any`	Allows forwarding of Layer 2 traffic if the conditions are matched. This creates an ACE for the ACL.
Step 5	**permit** {*src-mac mask* | **any**} {*dest-mac mask* | **any**} **Example:** `Device(config-ext-macl)# permit 00aa.bbcc.ddeb 0.0.0 any`	Allows forwarding of Layer 2 traffic if the conditions are matched. This creates an ACE for the ACL.
Step 6	**permit** {*src-mac mask* | **any**} {*dest-mac mask*} | **any**} **Example:**	Allows forwarding of Layer 2 traffic if the conditions are matched. This creates an ACE for the ACL.

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

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

Configuration Examples for Layer 2 Access Control Lists on EVCs

Example Applying a Layer 2 ACL to a Service Instance

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

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

 permit 00aa.bbcc.adec 0.0.0 any

 permit 00aa.bbcc.bdec 0.0.0 any

 permit 00aa.bbcc.cdec 0.0.0 any

 permit 00aa.bbcc.edec 0.0.0 any

 permit 00aa.bbcc.fdec 0.0.0 any

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

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

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

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

permit 00aa.bbcc.adec 0.0.0 any

permit 00aa.bbcc.bdec 0.0.0 any

permit 00aa.bbcc.cdec 0.0.0 any

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Device# show ethernet service instance id 100 interface gigabitethernet 3/0/1 detail`	

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

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

```
Device# show ethernet service instance id 100 interface gig3/0/1 detail
Service Instance ID: 100
L2 ACL (inbound): test-acl
Associated Interface: Gig3/0/1
Associated EVC: test
L2protocol drop
CEVlans:
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
L2 ACL permit count: 10255
L2 ACL deny count: 53
```

The table below describes the significant fields in the output.

Table 7: show ethernet service instance Field Descriptions

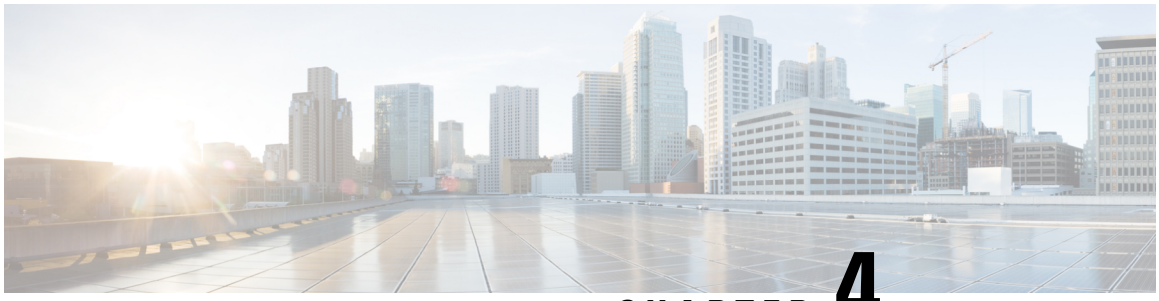
Field	Description
Service Instance ID	Displays the service instance ID.
L2 ACL (inbound):	Displays the ACL name.
Associated Interface:	Displays the interface details of the service instance.
Associated EVC:	Displays the EVC with which the service instance is associated.
CEVlans:	Displays details of the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

Example Displaying the Details of Configured Layer 2 ACL

The following sample output displays the details of a configured Layer 2 ACL.

```
Device# show access-lists
Extended IP access list ip-acl
10 permit ip any any
Extended MAC access list mac-acl
permit any any vlan 10
```

```
Device#
Device#sh access-lists mac-acl
Extended MAC access list mac-acl
permit any any vlan 10
```



CHAPTER 4

Configuring MAC Address Security on Service Instances and EVC Port Channels

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

Prerequisites for MAC Address Security on Service Instances and EVC Port Channels

- An understanding of service instances and bridge domains.
- An understanding of the concepts of MAC address limiting and how it is used for MAC security.
- An understanding of how port channels and EtherChannels work in a network.

Information About MAC Address Security on Service Instances and EVC Port Channels

Ethernet Virtual Circuits, Service Instances, and Bridge Domains

An Ethernet virtual circuit (EVC) as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint Layer 2 circuit. It is an end-to-end representation of a single instance of a Layer 2

service being offered by a provider to a customer. An EVC embodies the different parameters on which the service is being offered. A service instance is the instantiation of an EVC on a given port.

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

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

EVCs on Port Channels

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

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

EVCS uses the concepts of EVCs and service instances.

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

MAC Security and MAC Addressing

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

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

MAC Address Permit List

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

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

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

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

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

- If the address is already a denied address on the service instance, the configuration is rejected with an appropriate error message.
- If the acceptance of this address would increase the secure address count on the service instance beyond the maximum number allowed, an attempt is made to make room by removing an existing address from the MAC address table. The only candidate for removal is a dynamically learned address on the service instance. If sufficient room cannot be made, the configuration is rejected. If the acceptance of this address would increase the secure address count on the bridge domain beyond the maximum number allowed, an attempt is made to make room by removing an existing address from the MAC address table. The only candidate for removal is a dynamically learned address on the service instance. If room cannot be made, the configuration is rejected.

Note Default maximum address is '1' for a service instance.

- If the address is already permitted on another service instance in the same bridge domain, one of the following actions occur:
  - If the conflicting service instance has MAC security configured, the configuration is rejected with an appropriate error message.
  - If the conflicting service instance does not have MAC security configured, the configuration is accepted silently. (If the operator attempts to enable MAC security on the conflicting service instance, that attempt fails.)

MAC Address Deny List

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

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

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

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

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

Violation Response Configuration

A violation response is a response to a MAC security violation or a failed attempt to dynamically learn a MAC address due to an address violation. MAC security violations are of two types:

Type 1 Violation --The address of the ingress frame cannot be dynamically learned due to a deny list, or because doing so would cause the maximum number of secure addresses to be exceeded .

Type 2 Violation --The address of the ingress frame cannot be dynamically learned because it is already "present" on another secured service instance in the same bridge-domain.

There are three possible sets of actions that can be taken in response to a violation:

1. **Shutdown**
   - The ingress frame is dropped.
   - The service instance on which the offending frame arrived is shut down.
   - The event and the response are logged to SYSLOG.
2. **Restrict**
   - The ingress frame is dropped.
   - The event and the response are logged to SYSLOG.
3. **Protect**
   - The ingress frame is dropped.

Note The ingress frame is dropped silently, without sending any violation report to the SYSLOG.

Note The Restrict and Protect modes are applied on EFP level to discard the traffic. Both the modes are not appli on the Erroneous MAC level.

If a violation response is not configured, the default response mode is shutdown. The violation response can be configured to protect or restrict mode. A "no" form of a violation response, sets the violation response to the default mode of shutdown.

You are allowed to configure the desired response for a Type 1 and Type 2 violations on a service instance. For a Type 1 violation on a bridge domain (that is, if the learn attempt conforms to the policy configured on the service instance, but violates the policy configured on the bridge domain), the response is always "Protect." This is not configurable.

In shutdown mode, the service instance is put into the error disabled state immediate, an SNMP trap notification is transmitted, and a message is sent to the console and SYSLOG as shown below:

```
%ETHER_SERVICE-6-ERR_DISABLED:
Mac security violation - shutdown service instance 100 on interface gig 0/0/0
```

In Restrict mode, the violation report is sent to SYSLOG at level LOG_WARNING.

Support for the different types of violation responses depends on the capabilities of the platform. The desired violation response can be configured on the service instance. The configured violation response does not take effect unless and until MAC security is enabled using the **mac security** command.

MAC Address Aging Configuration

A specific time scheduler can be set to age out secured MAC addresses that are dynamically learned or statically configured on both service instances and bridge domains, thus freeing up unused addresses from the MAC address table for other active subscribers.

The set of rules applied to age out secured MAC addresses is called secure aging. By default, the entries in the MAC address table of a secured service instance are never aged out. This includes permitted addresses and dynamically learned addresses.

The **mac security aging time** *aging-time* command sets the aging time of the addresses in the MAC address table to <*n* > minutes. By default, this affects only dynamically learned (not including sticky) addresses--permitted addresses and sticky addresses are not affected by the application of this command.

By default, the aging time <n> configured via the **mac security aging time** *aging-time* command is an absolute time. That is, the age of the MAC address is measured from the instant that it was first encountered on the service instance. This interpretation can be modified by using the **mac security aging time** *aging-time* **inactivity** command, which specifies that the age <n> be measured from the instant that the MAC address was last encountered on the service instance.

The **mac security aging static**and **mac security aging sticky** commands specify that the **mac security aging time**aging-time command must be applicable to permitted and sticky MAC addresses, respectively. In the case of permitted MAC addresses, the absolute aging time is measured from the time the address is entered into the MAC address table (for example, when it is configured or whenever the **mac security** command is entered--whichever is later).

If the **mac security aging time** command is not configured, the **mac security aging static** command has no effect.

Sticky MAC Address Configurations

The ability to make dynamically learned MAC addresses on secured service instances permanent even after interface transitions or device reloads can be set up and configured. A dynamically learned MAC address that is made permanent on a secured service instance is called a “sticky MAC address”. The **mac security sticky** command is used to enable the sticky MAC addressing feature on a service instance.

With the “sticky” feature enabled on a secured service instance, MAC addresses learned dynamically on the service instance are kept persistent across service instance line transitions and device reloads.

The sticky feature has no effect on statically configured MAC addresses. The sticky addresses are saved in the running configuration. Before the device is reloaded, it is the responsibility of the user to save the running configuration to the startup configuration. Doing this will ensure that when the device comes on, all the MAC addresses learned dynamically previously are immediately populated into the MAC address table.

The **mac security sticky address** *mac-address* command can configure a specific MAC address as a sticky MAC address. The use of this command is not recommended for the user because configuring a MAC address as a static address does the same thing. When sticky MAC addressing is enabled by the **mac security sticky** command, the dynamically learned addresses are marked as sticky and a **mac security sticky address**

mac-address command is automatically generated and saved in the running configuration for each learned MAC address on the service instances.

Aging for Sticky Addresses

MAC addresses learned on a service instance that has the sticky behavior enabled are subject to aging as configured by the **mac security aging time** and **mac security aging sticky** commands. In other words, for the purpose of aging functionality, sticky addresses are treated the same as dynamically learned addresses.

Transitions

This section contains a description of the expected behavior of the different MAC security elements when various triggers are applied; for example, configuration changes or link state transitions.

MAC Security Enabled on a Service Instance

When MAC security is enabled on a service instance, all existing MAC table entries for the service instance are purged. Then, permitted MAC address entries and sticky addresses are added to the MAC table, subject to the prevailing MAC address limiting constraints on the bridge domain.

If MAC address limits are exceeded, any MAC address that fails to get added is reported via an error message to the console, the attempt to enable MAC security on the service instance fails, and the already added permitted entries are backed out or removed.

The aging timer for all entries is updated according to the secure aging rules.

MAC Security Disabled on a Service Instance

The existing MAC address table entries for this service instance are purged.

Service Instance Moved to a New Bridge Domain

This transition sequence applies to all service instances, whether or not they have MAC security configured. All the MAC addresses on this service instance in the MAC address table of the old bridge domain are removed. The count of dynamically learned addresses in the old bridge domain is decremented. Then, all the MAC security commands are permanently erased from the service instance.

Service Instance Removed from a Bridge Domain

All the MAC addresses in the MAC address table that attributable to this service instance are removed, and the count of dynamically learned addresses in the bridge domain is decremented. Since MAC security is applicable only on service instances that are members of a bridge domain, removing a service instance from a bridge domain causes all the MAC security commands to be erased permanently.

Service Instance Shut Down Due to Violation

All dynamically learned MAC addresses in the MAC address table are removed, and all the other MAC security state values are left unchanged. The only change is that no traffic is forwarded, and therefore no learning can take place.

Interface Service Instance Down Linecard OIR Removed

The MAC tables of all the affected bridge domains are cleared of all the entries attributable to the service instances that are down.

Interface Service Instance Re-activated Linecard OIR Inserted

The static and sticky address entries in the MAC tables of the affected bridge domains are re-created to the service instances that are activated.

MAC Address Limit Decreased

When the value of the MAC address limit on the service instance is changed initially, a sanity check is performed to ensure that the new value of <n> is greater than or equal to the number of permitted entries. If not, the command is rejected. The MAC table is scanned for addresses that are attributable to this service instance, and dynamically learned MAC addresses are removed when the new MAC address limit is less than the old MAC address limit.

Sticky Addresses Added or Removed on a Service Instance

Existing dynamically learned MAC addresses remain unchanged. All new addresses learned become “sticky” addresses.

Disabling sticky addresses causes all sticky secure MAC addresses on the service instance to be removed from the MAC address table. All new addresses learned become dynamic addresses on the service instance and are subject to aging.

How to Configure MAC Address Limiting on Service Instances Bridge Domains and EVC Port Channels

Enabling MAC Security on a Service Instance

Perform this task to enable MAC address security on a service instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 8	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Enabling MAC Security on an EVC Port Channel

Before you begin

Note

- Bridge-domain, xconnect, and Ethernet virtual circuits (EVCs) are allowed only over the port channel interface and the main interface.
- If you configure a physical port as part of a channel group, you cannot configure EVCs under that physical port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *channel-group*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security**
8. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-group* **Example:** `Device(config)# interface port-channel 2`	Specifies the port channel group number and enters interface configuration mode. • Acceptable values are integers from 1 to 64.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance on an interface and enters service instance configuration mode.

	Command or Action	Purpose
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 8	end **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Configuring a MAC Address Permit List

Perform this task to configure permitted MAC addresses on a service instance that is a member of a bridge domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security address permit** *mac-address*
8. **mac security address permit** *mac-address*
9. **mac security address permit** *mac-address*
10. **mac security address permit** *mac-address*
11. **mac security address permit** *mac-address*
12. **mac security**
13. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used for mapping ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaaa`	Adds the specified MAC address as a permit MAC address for the service instance.
Step 8	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaab`	Adds the specified MAC address as a permitted MAC address for the service instance.
Step 9	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaac`	Adds the specified MAC address as a permitted MAC address for the service instance.

	Command or Action	Purpose
Step 10	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaad`	Adds the specified MAC address as a permitted MAC address for the service instance.
Step 11	**mac security address permit** *mac-address* **Example:** `Device(config-if-srv)# mac security address permit a2aa.aaaa.aaae`	Adds the specified MAC address as a permitted MAC address for the service instance.
Step 12	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 13	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Configuring a MAC Address Deny List

Perform this task to configure a list of MAC addresses that are not allowed on a service instance that is a member of a bridge domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security address deny** *mac-address*
8. **mac security address deny** *mac-address*
9. **mac security address deny** *mac-address*
10. **mac security address deny** *mac-address*
11. **mac security address deny** *mac-address*
12. **mac security**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaaa`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 8	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaab`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 9	**mac security address deny** *mac-address* **Example:**	Adds the specified MAC address as a denied MAC address for the service instance.

	Command or Action	Purpose
	`Device(config-if-srv)# mac security address deny a2aa.aaaa.aaac`	
Step 10	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaad`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 11	**mac security address deny** *mac-address* **Example:** `Device(config-if-srv)# mac security address deny a2aa.aaaa.aaae`	Adds the specified MAC address as a denied MAC address for the service instance.
Step 12	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 13	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Configuring MAC Address Security on a Service Instance

Perform this task to configure an upper limit for the number of secured MAC addresses allowed on a service instance. This number includes addresses added as part of a permit list as well as dynamically learned MAC addresses. If the upper limit is decreased, all learned MAC entries are removed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security maximum addresses** *maximum-addresses*
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security maximum addresses** *maximum-addresses* **Example:** `Device(config-if-srv)# mac security maximum addresses 500`	Sets the maximum number of secure addresses permitted on the service instance. **Note** Default value for a service instance is '1'.
Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Configuring a MAC Address Violation

Perform this task to specify the expected behavior of a device when an attempt to dynamically learn a MAC address fails because the configured MAC security policy on the service instance was violated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. Do one of the following:
   - **mac security violation restrict**
   - **mac security violation protect**
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.

	Command or Action	Purpose
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.

Step 7	Do one of the following:
	• **mac security violation restrict**
	• **mac security violation protect**
	Example:
	`Device(config-if-srv)# mac security violation restrict`
	Example:
	`Device(config-if-srv)# mac security violation protect`
		Sets the violation mode (for Type 1 and 2 violations) to restrict.
		or
		Sets the violation mode (for Type 1 and 2 violations) to protect.
		• If a MAC security violation response is not specified, by default, the violation mode is shutdown.

Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Configuring MAC Address Aging

Perform this task to configure the aging of secured MAC addresses under MAC security. Secured MAC addresses are not subject to the normal aging of MAC table entries. If aging is not configured, secured MAC addresses are never aged out.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security aging time** *aging-time* **[inactivity]**
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security aging time** *aging-time* **[inactivity]** **Example:** `Device(config-if-srv)# mac security aging time 200 inactivity`	Sets the aging time for secure addresses, in minutes. The optional **inactivity** keyword specifies that the aging out of addresses is based on inactivity of the sending hosts (as opposed to absolute aging).
Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Configuring a Sticky MAC Address

If sticky MAC addressing is configured on a secured service instance, MAC addresses that are learned dynamically on the service instance are retained during a link-down condition. Perform this task to configure sticky MAC addresses on a service instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security sticky address** *mac-address*
8. **mac security**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 100`	Defines the matching criteria to be used to map ingress dot1q frames on an interface to the appropriate service instance.

	Command or Action	Purpose
Step 6	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 200`	Binds the service instance to a bridge- domain instance where *bridge-id* is the identifier for the bridge- domain instance.
Step 7	**mac security sticky address** *mac-address* **Example:** `Device(config-if-srv)# mac security sticky address 1111.2222.3333`	Sets up a MAC address to be declared as a sticky MAC address on the service instance.
Step 8	**mac security** **Example:** `Device(config-if-srv)# mac security`	Enables MAC security on the service instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Returns to user EXEC mode.

Displaying the MAC Security Status of a Specific Service Instance

Perform this task to display the MAC security status of a service instance.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type* *number* **mac security**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type* *number* **mac security** **Example:** `Device# show ethernet service instance id 100 interface gigabitethernet1/1 mac security`	Displays the MAC security status of a specific service instance.

	Command or Action	Purpose
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Displaying the Service Instances with MAC Security Enabled

Perform this task to display all the service instances with MAC security enabled.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance mac security**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance mac security** **Example:** `Device# show ethernet service instance mac security`	Displays all the service instances with MAC security enabled.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Displaying the Service Instances with MAC Security Enabled on a Specific Bridge Domain

Perform this task to display the service instances on a specific bridge domain that have MAC security enabled.

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *id* **mac security**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bridge-domain** *id* **mac security** **Example:** `Device# show bridge-domain 100 mac security`	Displays all the service instances with MAC security enabled on a specific bridge domain.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Showing the MAC Addresses of All Secured Service Instances

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance mac security address**
3. **show mac address-table secure**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance mac security address** **Example:** `Device# show ethernet service instance mac security address`	Displays the secured addresses on all the service instances.
Step 3	**show mac address-table secure** **Example:** `Device# show mac address-table secure`	Displays the secure MAC address on the service instances.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Showing the MAC Addresses of a Specific Service Instance

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type number* **mac security address**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type number* **mac security address** **Example:** `Device# show ethernet service instance id 200 interface GigabitEthernet 1/0 mac security address`	Displays the addresses of a specific service instance.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Showing the MAC Addresses of All Service Instances on a Specific Bridge Domain

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *id* **mac security address**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bridge-domain** *id* **mac security address** **Example:** `Device# show bridge-domain 100 mac security address`	Displays the secured addresses of all the service instances on a specified bridge domain.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Showing the MAC Security Statistics of a Specific Service Instance

This section describes how to display the MAC security statistics of a specific service instance.

SUMMARY STEPS

1. **enable**
2. **show ethernet service instance id** *id* **interface** *type number* **mac security statistics**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ethernet service instance id** *id* **interface** *type number* **mac security statistics** **Example:** `Device# show ethernet service instance id 100 interface gigabitethernet1/1 mac security statistics`	Displays the MAC security statistics of a specific service instance.
Step 3	**end** **Example:**	Returns to user EXEC mode.

	Command or Action	Purpose
	`Device# end`	

Showing the MAC Security Statistics of All Service Instances on a Specific Bridge Domain

Perform this task to display the MAC security statistics of all the service instances on a specific bridge domain.

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *bridge-id* **mac security statistics**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bridge-domain** *bridge-id* **mac security statistics** **Example:** `Device# show bridge-domain 100 mac security statistics`	Displays the MAC security statistics of all service instances that belong to a specific bridge domain.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Showing the Last Violation Recorded on Each Service Instance on a Specific Bridge Domain

Perform this task to display the last violation recorded on each service instance on a specific bridge domain. Service instances on which there have been no violations are excluded from the output.

SUMMARY STEPS

1. **enable**
2. **show bridge-domain** *bridge-id* **mac security last violation**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bridge-domain** *bridge-id* **mac security last violation** **Example:** `Device# show bridge-domain 100 mac security last violation`	Displays information about the last violation recorded on each of the service instances that belong to the bridge domain.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Clearing All Dynamically Learned Secure MAC Addresses on a Service Instance

Perform this task to clear all dynamically learned Secure MAC addresses on a service instance.

SUMMARY STEPS

1. **enable**
2. **clear ethernet service instance id** *id* **interface** *type number* **mac table**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ethernet service instance id** *id* **interface** *type number* **mac table** **Example:** `Device# clear ethernet service instance id 100 interface gigabitethernet0/0/1 mac table`	Clears all the dynamically learned Secure MAC addresses on the specified service instance.

	Command or Action	Purpose
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

Clearing All Dynamically Learned MAC Addresses on a Bridge Domain

Perform this task to clear all dynamically learned MAC addresses on a bridge domain.

SUMMARY STEPS

1. **enable**
2. **clear bridge-domain** *bridge-id* **mac table**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bridge-domain** *bridge-id* **mac table** **Example:** `Device# clear bridge-domain 100 mac table`	Clears all dynamically learned MAC addresses on the specified bridge domain.
Step 3	**end** **Example:** `Device# end`	Returns to user EXEC mode.

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

Example Enabling MAC Security on a Service Instance

The following example shows how to enable MAC security on a service instance:

```
Device> enable
Device# configure terminal
```

```
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Enabling MAC Security on an EVC Port Channel

The following example shows how to enable MAC Security on an EVC port channel:

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a MAC Address Permit List

The following example shows how to configure a MAC address permit list:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security maximum addresses 5
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaaa
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaab
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaac
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaad
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaae
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a MAC Address Deny List

The following example shows how to configure a MAC address deny list:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaaa
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaab
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaac
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaad
Device(config-if-srv)# mac security address deny a2aa.aaaa.aaae
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a MAC Address Security on a Service Instance

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security maximum addresses 10
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a MAC Address Violation Response

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security address permit a2aa.aaaa.aaaa
Device(config-if-srv)# mac security violation protect
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring MAC Address Aging

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 4/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security aging time 10
Device(config-if-srv)# mac security
Device(config-if-srv)# end
```

Example Configuring a Sticky MAC Address

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 3/0/1
Device(config-if)# service instance 100 ethernet
Device(config-if-srv)# encapsulation dot1Q 100
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# mac security sticky address 1111.2222.3333
Device(config-if-srv)# mac security
```

Example Displaying the MAC Addresses on a Specific Secure Service Instance

```
Device# show ethernet service instance id 10 inter gig 0/0/3 mac security
```

```
address Bridge-domain 10

MAC Address       Type
0000.00ac.ef02    sticky
0000.00ac.ef03    sticky
0001.0001.aaaa    dynamic
0001.0001.aaab    dynamic
```

Example Displaying the Last Violation on a Specific Service Instance

```
Device# show bridge-domain 100 mac security last violation Te0/0/3 ServInst 200
Last violation at: 15:54:25 IST Fri Jun 5 2015
Source MAC address: 0000.1111.1111
Reason: Re-learn attempt
Total violation count: 321
```

Example Displaying the MAC Security Status of a Specific Service Instance

```
Device# show ethernet service instance id 100 interface te0/0/3 mac security
Bridge-domain 100
MAC Security enabled: yes
```

Example Displaying the MAC Addresses of All Secured Service Instances

```
Device# show ethernet service instance mac security address
Port                   Bridge-domain    MAC Address       Type
Gi0/0/3 ServInst 10    10  0000.00ac.ef02 sticky
Gi0/0/3 ServInst 10    10   0000.00ac.ef03 sticky
Gi0/0/3 ServInst 10    10   0000.00ac.ef04  dynamic
Gi0/0/3 ServInst 10    10   0000.00ac.ef05  dynamic
Gi0/0/3 ServInst 10    10   0000.00ac.ef06  sticky
Gi0/0/3 ServInst 10    10   0000.00ac.ef07  dynamic
Gi0/0/3 ServInst 10    10   0000.00ac.ef08  dynamic
Gi0/0/3 ServInst 10    10   0000.00ac.ef09  dynamic
Gi0/0/3 ServInst 10    10   0000.00ac.ef0a  dynamic
Gi0/0/3 ServInst 10    10   0000.00ac.ef0b  dynamic
```

Example Displaying the MAC Security Statistics of All Service Instances

In the following example, the numbers of allowed and actual secured addresses recorded on the service instance are displayed.

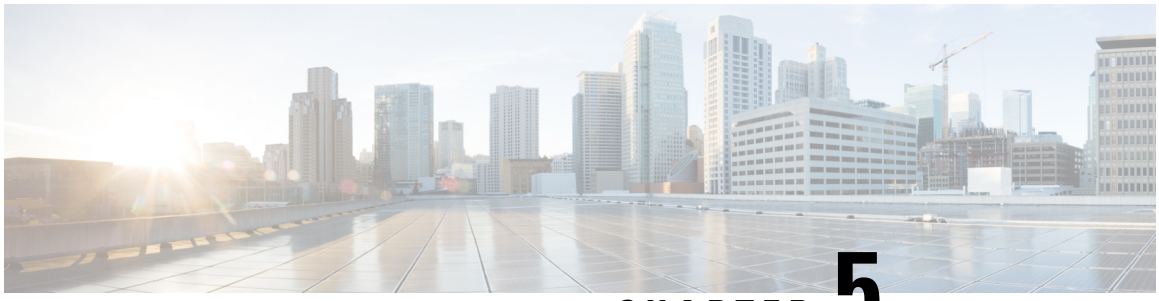
```
Device#  show ethernet serv instance mac security statistics
Te0/0/3 ServInst 100 (bridge-domain 100)
Current secure addresses: 1
Permitted addresses: 10
Te0/0/3 ServInst 200 (bridge-domain 100)
Current secure addresses: 0
Permitted addresses: 1
Te0/0/3 ServInst 300 (bridge-domain 100)
Current secure addresses: 0
Permitted addresses: 1
```

Example: Displaying the MAC Addresses on All Service Instances for a Bridge Domain

```
Router# show bridge-domain 10 mac security address
Port                  MAC Address       Type
Gi0/0/3 ServInst 10   0000.00ac.ef02    sticky
Gi0/0/3 ServInst 10   0000.00ac.ef03    sticky
Gi0/0/3 ServInst 10   0000.00ac.ef04    dynamic
Gi0/0/3 ServInst 10   0000.00ac.ef05    dynamic
Gi0/0/3 ServInst 10   0000.00ac.ef06    sticky
Gi0/0/3 ServInst 10   0000.00ac.ef07    dynamic
Gi0/0/3 ServInst 10   0000.00ac.ef08    dynamic
Gi0/0/3 ServInst 10   0000.00ac.ef09    dynamic
Gi0/0/3 ServInst 10   0000.00ac.ef0a    dynamic
Gi0/0/3 ServInst 10   0000.00ac.ef0b    dynamic
```

Example Displaying the Secured Service Instances for a Specific Bridge Domain

```
Router# show bridge-domain 10 mac security
Gi0/0/3 ServInst 10
MAC Security enabled: yes
```



CHAPTER 5

Static MAC Address Support on Service Instances

The Multicast and Unicast static MAC address support on Service Instances feature supports configuration of a static MAC address on a pseudoport. Use of a static MAC address for Broadband Network Gateway (BNG) upstream traffic enables traffic forwarding while conserving MAC table resources and limiting the traffic flood by creating multicast groups.

Prerequsites for Static MAC Address Support on Service Instances

- Knowledge of both port and bridge domain limitations.
- Knowledge of service instances.

Restrictions for Static MAC Address Support on Service Instances

- Static MAC configuration is *not* allowed at secure service instance.
- Static MAC addresses are programmed only on switch processors (both active and standby).
- The Static MAC address on Pseudowires is *not* supported on the Cisco ASR 900 Series Routers.
- Static MAC address configuration is *not* supported on Trunk EFP.

Information about Static MAC Address Support on Service Instances

Static MAC address configuration on service instances eliminates the need for MAC address learning, which is required for traffic forwarding. In the upstream direction, without MAC address learning, MAC address table resources can be conserved and network resources optimized.

When a bridge domain ID is either changed or deleted for a service instance, all static MAC addresses are removed.

When a service instance is deleted, all static MAC addresses on that pseudoport are removed.

Benefits of Static MAC Address Support on Service Instances

- Facilitates optimization of network resources
- Conserves MAC table resources when used for upstream traffic

Configuring a Static MAC Address on a Service Instance

Perform this task to manually configure a static MAC address on a service instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet** [*evc-id*]
5. **encapsulation dot1q** *vlan-id* [**,** *vlan-id*[- *vlan-id*]] [**native**]
6. **bridge-domain** *bridge-id* [**split-horizon**[**group** *group-id*]]
7. **mac static address** *mac-addr* [**auto-learn**]
8. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/2/1`	Configures an interface type and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** [*evc-id*] **Example:** `Router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance on an interface and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* [**,** *vlan-id*[**-** *vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 6	**bridge-domain** *bridge-id* [**split-horizon**[**group** *group-id*]] **Example:** `Router(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 7	**mac static address** *mac-addr* [**auto-learn**] **Example:** `Router(config-if-srv)# mac static address 0000.bbbb.cccc`	Configures a static MAC address.
Step 8	**exit** **Example:** `Router(config-if-srv)# exit`	Returns the CLI to privileged EXEC mode.

Example for Configuring a Static MAC Address on a Service Instance

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet 0/2/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac static address 0000.bbbb.cccc
Router(config-if-srv)# exit
```

Verifying Configured Static MAC Addresses on a Service Instance

Use one or more of the following commands to verify the configured static MAC address on a service instance:

- **show bridge-domain**
- **show mac-address-table**

Example: Verifying Configured Static MAC Addresses on a Service Instance

show bridge-domain

The sample output for the **show bridge-domain** command:

```
Router# show bridge-domain 10 mac static address

Bridge-Domain ID : 10
Static MAC count : System : 1, bridge-domain : 1

Port                                    Address          Action
Gi0/3/7 ServInst 10                     aaa1.123c.bc32
```

show mac-address-table

The sample output for the **show mac-address-table** command:

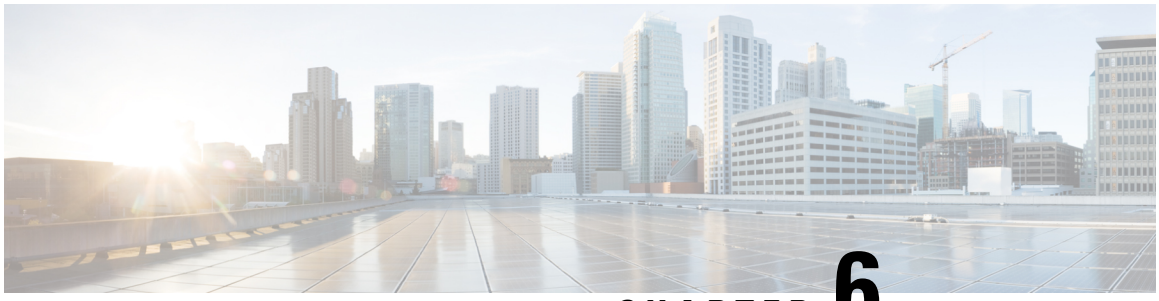
```
Router# show mac-address-table bdomain 10

   Nile Mac Address Entries

   BD    mac addr        type     ports

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

   10    aaa1.123c.bc32  STATIC   Gi0/3/7.Efp10
```



CHAPTER 6

MAC Limiting

This document describes how to configure MAC limiting.

Restrictions and Usage Guidelines

MAC limiting is supported on the following interface types:

- You can apply MAC limiting only to bridge-domains.
- MAC limiting is supported for dynamic MAC addresses.

Configuring MAC Limiting

Mac address limiting per bridge-domain restricts the number of MAC addresses that the router learns in bridge-domain on an EFP, pseudowire or switchport.

Note Local connect feature is not supported on the Cisco router. However, to simulate a local connect scenario, configure the connecting EFPs on the same bridge domain and disable the mac-learning on the bridge domain by setting the MAC limit to 0. Use the **mac-address-table limit bdomain** *num* **maximum** *0* **action limit** command to disable mac-learning on the router.

When the total number of addresses in a bridge-domain exceeds the maximum number, the router takes a violation action. You can enable the following actions:

- Warning—The router sends a syslog message and takes no further action. The router continues learning new MAC addresses and forwarding traffic.
- Limit—The router sends a syslog message and generates a trap; MAC learning is disabled on the bridge-domain until the recovery mechanism activates. Flooding of frames with new MAC addresses continues; to disable flooding, use the flood keyword. Flooding continues once the total number of MAC entries drops below the threshold value. This option applies only when you configure the limit keyword.

Note The threshold value must be 80% of the maximum value configured for the recovery mechanism.

- Shutdown—If the number of addresses exceeds the maximum (MAX) value, the router sends a syslog message and moves the bridge-domain (bdomain) to a disabled state. To restore the bridge-domain, disable and re-enable the mac-limiting feature.

Note **Warning** is the default action when no action is configured.

Note The functionality of automatic error recovery is *not* supported on the Cisco ASR 900 RSP2 module.

Before you begin

SUMMARY STEPS

1. **configure terminal**
2. mac-address-table limit *bdomain id maximum num action {warning | limit | shutdown} [flood]*
3. **end**
4. **show mac-address-table limit [bdomain id]**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	mac-address-table limit *bdomain id maximum num action {warning | limit | shutdown} [flood]*	Sets the specific limit and any optional actions to be imposed at the bridge-domain level. The default **maximum** value is 500.
Step 3	**end**	Return to privileged EXEC mode.
Step 4	**show mac-address-table limit [bdomain id]**	Displays the information about the MAC-address table.
Step 5	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Example of Enabling Per-Bridge-Domain MAC Limiting

This example shows how to enable per-bridge-domain MAC limiting.

```
Router# enable
Router# configure terminal
Router(config)# mac-address-table limit bdomain 10 maximum 100 action limit flood
Router(config)# end
```

```
Router#show mac-address-table limit bdomain 10
  bdomain       action      flood       maximum     Total entries     Current state
-------------+----------+------------+------------+---------------+---------------
   10           limit      Disable        100            0             Within Limit
```