Preface

The *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide* provides information and procedures related to router interface and hardware configuration.

The preface contains the following sections:

- Changes to This Document
- Obtaining Documentation and Submitting a Service Request

Changes to This Document

Table 1 lists the technical changes made to this document since it was first printed.

Table 1 Changes to This Document

Revision	Date	Change Summary
OL-26061-03	September 2012	Republished with documentation updates for Cisco IOS XR Release 4.2.3 features.
OL-26061-02	June 2012	Republished with documentation updates for Cisco IOS XR Release 4.2.1 features.
OL-26061-01	December 2011	Initial release of this document.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

Preconfiguring Physical Interfaces on the Cisco ASR 9000 Series Router

This module describes the preconfiguration of physical interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

Preconfiguration is supported for the following types of interfaces and controllers:

- Gigabit Ethernet
- 10-Gigabit Ethernet
- Management Ethernet
- Packet-over-SONET/SDH (POS)
- Serial
- SONET controllers and channelized SONET controllers

Preconfiguration allows you to configure modular services cards before they are inserted into the router. When the cards are inserted, they are instantly configured.

The preconfiguration information is created in a different system database tree (known as the *preconfiguration directory* on the route switch processor [RSP]), rather than with the regularly configured interfaces.

There may be some preconfiguration data that cannot be verified unless the modular services card is present, because the verifiers themselves run only on the modular services card. Such preconfiguration data is verified when the modular services card is inserted and the verifiers are initiated. A configuration is rejected if errors are found when the configuration is copied from the preconfiguration area to the active area.

Note Only physical interfaces can be preconfigured.

Feature History for Preconfiguring Physical Interfaces

Release	Modification
Release 3.7.2	Ethernet interface preconfiguration was introduced.
Release 4.0.0	POS interface preconfiguration was introduced.

Contents

Prerequisites for Preconfiguring Physical Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before preconfiguring physical interfaces, be sure that the following conditions are met:

- Preconfiguration drivers and files are installed. Although it may be possible to preconfigure physical interfaces without a preconfiguration driver installed, the preconfiguration files are required to set the interface definition file on the router that supplies the strings for valid interface names.

Information About Preconfiguring Physical Interfaces

To preconfigure interfaces, you must understand the following concepts:

- Physical Interface Preconfiguration Overview, page 2
- Benefits of Interface Preconfiguration, page 3
- Use of the Interface Preconfigure Command, page 3
- Active and Standby RSPs and Virtual Interface Configuration, page 4

Physical Interface Preconfiguration Overview

Preconfiguration is the process of configuring interfaces before they are present in the system. Preconfigured interfaces are not verified or applied until the actual interface with the matching location (rack/slot/module) is inserted into the router. When the anticipated modular services card is inserted and the interfaces are created, the precreated configuration information is verified and, if successful, immediately applied to the router’s running configuration.

Note When you plug the anticipated modular services card in, make sure to verify any preconfiguration with the appropriate **show** commands.

Use the **show run** command to see interfaces that are in the preconfigured state.

Note We recommend filling out preconfiguration information in your site planning guide, so that you can compare that anticipated configuration with the actual preconfigured interfaces when that card is installed and the interfaces are up.

Tip Use the **commit best-effort** command to save the preconfiguration to the running configuration file. The **commit best-effort** command merges the target configuration with the running configuration and commits only valid configuration (best effort). Some configuration might fail due to semantic errors, but the valid configuration still comes up.

Benefits of Interface Preconfiguration

Preconfigurations reduce downtime when you add new cards to the system. With preconfiguration, the new modular services card can be instantly configured and actively running during modular services card bootup.

Another advantage of performing a preconfiguration is that during a card replacement, when the modular services card is removed, you can still see the previous configuration and make modifications.

Use of the Interface Preconfigure Command

Interfaces that are not yet present in the system can be preconfigured with the **interface preconfigure** command in global configuration mode.

The **interface preconfigure** command places the router in interface configuration mode. Users should be able to add any possible interface commands. The verifiers registered for the preconfigured interfaces verify the configuration. The preconfiguration is complete when the user enters the **end** command, or any matching **exit** or global configuration mode command.

Note It is possible that some configurations cannot be verified until the modular services card is inserted.

Note Do not enter the **no shutdown** command for new preconfigured interfaces, because the **no** form of this command removes the existing configuration, and there is no existing configuration.

Users are expected to provide names during preconfiguration that will match the name of the interface that will be created. If the interface names do not match, the preconfiguration cannot be applied when the interface is created. The interface names must begin with the interface type that is supported by the router and for which drivers have been installed. However, the slot, port, subinterface number, and channel interface number information cannot be validated.

Note Specifying an interface name that already exists and is configured (or an abbreviated name like e0/3/0/0) is not permitted.

Active and Standby RSPs and Virtual Interface Configuration

The standby RSP is available and in a state in which it can take over the work from the active RSP should that prove necessary. Conditions that necessitate the standby RSP to become the active RSP and assume the active RSP's duties include:

- Failure detection by a watchdog
- Standby RSP is administratively commanded to take over
- Removal of the active RSP from the chassis

If a second RSP is not present in the chassis while the first is in operation, a second RSP may be inserted and will automatically become the standby RSP. The standby RSP may also be removed from the chassis with no effect on the system other than loss of RSP redundancy.

After failover, the virtual interfaces will all be present on the standby (now active) RSP. Their state and configuration will be unchanged, and there will have been no loss of forwarding (in the case of tunnels) over the interfaces during the failover. The Cisco ASR 9000 Series Router uses nonstop forwarding (NSF) over tunnels through the failover of the host RSP.

Note The user does not need to configure anything to guarantee that the standby interface configurations are maintained.

How to Preconfigure Physical Interfaces

This task describes only the most basic preconfiguration of an interface.

SUMMARY STEPS

1. **configure**
2. **interface preconfigure** *type interface-path-id*
3. **ipv4 address** *ip-address subnet-mask*
4. Configure additional interface parameters.
5. **end**
   or
   commit
6. **exit**
7. **exit**
8. **show running-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface preconfigure`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface preconfigure GigabitEthernet 0/1/0/0`
		Enters interface preconfiguration mode for an interface, where *type* specifies the supported interface type that you want to configure and *interface-path-id* specifies the location where the interface will be located in *rack***/***slot***/***module***/***port* notation.

Step 3	**`ipv4 address`** *`ip-address subnet-mask`*
	`or`
	`ipv4 address` *`ip-address`***`/`***`prefix`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-pre)# ipv4 address 192.168.1.2/32`
		Assigns an IP address and mask to the interface.

Step 4	Configure additional interface parameters, as described in this manual in the configuration chapter that applies to the type of interface that you are configuring.	

	Command or Action	Purpose

Step 5	**end**
	or
	commit best-effort
	Example:
	`RP/0/RSP0/CPU0:router(config-if-pre)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if-pre)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit best-effort** command to save the configuration changes to the running configuration file and remain within the configuration session. The **commit best-effort** command merges the target configuration with the running configuration and commits only valid changes (best effort). Some configuration changes might fail due to semantic errors.

Step 6	**show running-config** **Example:** `RP/0/RSP0/CPU0:router# show running-config`	(Optional) Displays the configuration information currently running on the router.

Configuration Examples for Preconfiguring Physical Interfaces

This section contains the following example:

Preconfiguring an Interface: Example, page 6

Preconfiguring an Interface: Example

The following example shows how to preconfigure a basic Ethernet interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface preconfigure GigabitEthernet 0/1/0/0
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.2/32
RP/0/RSP0/CPU0:router(config-if)# commit
```

Additional References

The sections that follow provide references related to the preconfiguration of physical interfaces.

Related Documents

Related Topic	Document Title
Master command reference	*Cisco ASR 9000 Series Aggregation Services Routers Master Command Listing*
Interface configuration commands	*Cisco ASR 9000 Series Aggregation Services Routers Interface and Hardware Component Command Reference*
Initial system bootup and configuration information	*Cisco ASR 9000 Series Router Getting Started Guide*
Information about user groups and task IDs	*Cisco IOS XR Task ID Reference Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Advanced Configuration and Modification of the Management Ethernet Interface on the Cisco ASR 9000 Series Router

This module describes the configuration of Management Ethernet interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

Before you can use Telnet to access the router through the LAN IP address, you must set up a Management Ethernet interface and enable Telnet servers, as described in the *Configuring General Router Features* module of the *Cisco ASR 9000 Series Router Getting Started Guide*. This module describes how to modify the default configuration of the Management Ethernet interface after it has been configured, as described in the *Cisco ASR 9000 Series Router Getting Started Guide.*

Note Forwarding between physical layer interface modules (PLIM) ports and Management Ethernet interface ports is disabled by default. To enable forwarding between PLIM ports and Management Ethernet interface ports, use the **rp mgmtethernet forwarding** command.

Note Although the Management Ethernet interfaces on the system are present by default, the user must configure these interfaces to use them for accessing the router, using protocols and applications such as Simple Network Management Protocol (SNMP), Common Object Request Broker Architecture (CORBA), HTTP, extensible markup language (XML), TFTP, Telnet, and command-line interface (CLI).

Feature History for Configuring Management Ethernet Interfaces

Release	Modification
Release 3.7.2	This feature was introduced on the Cisco ASR 9000 Series Router.

Contents

Prerequisites for Configuring Management Ethernet Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before performing the Management Ethernet interface configuration procedures that are described in this chapter, be sure that the following tasks and conditions are met:

- You have performed the initial configuration of the Management Ethernet interface, as described in the *Configuring General Router Features* module of the *Cisco ASR 9000 Series Router Getting Started Guide*.
- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.
- You know how to apply the generalized interface name specification *rack/slot/module/port*.

  For further information on interface naming conventions, refer to the *Cisco ASR 9000 Series Router Getting Started Guide*.

Note For transparent switchover, both active and standby Management Ethernet interfaces are expected to be physically connected to the same LAN or switch.

Information About Configuring Management Ethernet Interfaces

To configure Management Ethernet interfaces, you must understand the following concept:

- Default Interface Settings, page 10

Default Interface Settings

Table 2 describes the default Management Ethernet interface settings that can be changed by manual configuration. Default settings are not displayed in the **show running-config** command output.

Table 2 Management Ethernet Interface Default Settings

Parameter	Default Value	Configuration File Entry
Speed in Mbps	Speed is autonegotiated.	**speed** [**10** | **100** | **1000**] To return the system to autonegotiate speed, use the **no speed** [**10** | **100** | **1000**] command.

Table 2 Management Ethernet Interface Default Settings

Parameter	Default Value	Configuration File Entry
Duplex mode	Duplex mode is autonegotiated.	**duplex** {**full** | **half**} To return the system to autonegotiated duplex operation, use the **no duplex {full | half}** command, as appropriate.
MAC address	MAC address is read from the hardware burned-in address (BIA).	**mac-address** *address* To return the device to its default MAC address, use the **no mac-address** *address* command.

How to Perform Advanced Management Ethernet Interface Configuration

This section contains the following procedures:

- Configuring a Management Ethernet Interface, page 11 (required)
- Configuring the Duplex Mode for a Management Ethernet Interface, page 13 (optional)
- Configuring the Speed for a Management Ethernet Interface, page 14 (optional)
- Modifying the MAC Address for a Management Ethernet Interface, page 16 (optional)
- Verifying Management Ethernet Interface Configuration, page 17 (optional)

Configuring a Management Ethernet Interface

Perform this task to configure a Management Ethernet interface. This procedure provides the minimal configuration required for the Management Ethernet interface.

The MTU is not configurable for the Management Ethernet Interface. The default value is 1514 bytes.

Note You do not need to perform this task if you have already set up the Management Ethernet interface to enable telnet servers, as described in the "*Configuring General Router* Features" module of the *Cisco ASR 9000 Series Router Getting Started Guide*.

SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **ipv4 address** *ip-address mask*
4. **no shutdown**
5. **end**
   or
   commit
6. **show interfaces MgmtEth** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface MgmtEth`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface MgmtEth 0/RSP0/CPU0/0`
		Enters interface configuration mode and specifies the Ethernet interface name and notation *rack*/*slot*/*module*/*port*.
		The example indicates port 0 on the RSP card that is installed in slot 0.

Step 3	**`ipv4 address`** *`ip-address mask`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224`
		Assigns an IP address and subnet mask to the interface.
		• Replace *ip-address* with the primary IPv4 address for the interface.
		• Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways:
		– The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address.
		– The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

Step 4	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration, which removes the forced administrative down on the interface, enabling it to move to an up or down state.

	Command or Action	Purpose

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**show interfaces MgmtEth** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show interfaces MgmtEth 0/RSP0/CPU0/0`	(Optional) Displays statistics for interfaces on the router.

Configuring the Duplex Mode for a Management Ethernet Interface

Perform this task to configure the duplex mode of the Management Ethernet interfaces for the RPs.

SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **duplex** [**full** | **half**]
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface MgmtEth`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface MgmtEth 0/RSP0/CPU0/0`
		Enters interface configuration mode and specifies the Management Ethernet interface name and instance.

Step 3	**`duplex`** `[`**`full`** `|` **`half`**`]`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# duplex full`
		Configures the interface duplex mode. Valid options are **full** or **half**.
		Note To return the system to autonegotiated duplex operation, use the **no duplex** command.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Speed for a Management Ethernet Interface

Perform this task to configure the speed of the Management Ethernet interfaces for the RPs.

SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **speed** {**10** | **100** | **1000**}

4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface MgmtEth`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface MgmtEth 0/RSP0/CPU0/0`
		Enters interface configuration mode and specifies the Management Ethernet interface name and instance.

Step 3	**`speed`** `{`**`10`** `|` **`100`** `|` **`1000`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# speed 100`
		Configures the interface speed parameter.
		On a Cisco ASR 9000 Series Router, valid **speed** options are **10** or **100** Mbps.
		Note The default Management Ethernet interface speed is autonegotiated.
		Note To return the system to the default autonegotiated speed, use the **no speed** command.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Modifying the MAC Address for a Management Ethernet Interface

Perform this task to configure the MAC layer address of the Management Ethernet interfaces for the RPs.

SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **mac-address** *address*
4. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface MgmtEth`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface MgmtEth 0/RSP0/CPU0/0`
		Enters interface configuration mode and specifies the Management Ethernet interface name and instance.

	Command or Action	Purpose

Step 3	**`mac-address`** *`address`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# mac-address 0001.2468.ABCD`
		Configures the MAC layer address of the Management Ethernet interface.
		Note To return the device to its default MAC address, use the **no mac-address** *address* command.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Verifying Management Ethernet Interface Configuration

Perform this task to verify configuration modifications on the Management Ethernet interfaces for the RPs.

SUMMARY STEPS

1. **show interfaces MgmtEth** *interface-path-id*
2. **show running-config**

Step 1	**`show interfaces MgmtEth`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show interfaces MgmtEth 0/RSP0/CPU0/0`
		Displays the Management Ethernet interface configuration.

Step 2	**`show running-config interface MgmtEth`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show running-config interface MgmtEth 0/RSP0/CPU0/0`
		Displays the running configuration.

Configuration Examples for Management Ethernet Interfaces

This section provides the following configuration examples:

- Configuring a Management Ethernet Interface: Example, page 18

Configuring a Management Ethernet Interface: Example

This example displays advanced configuration and verification of the Management Ethernet interface on the RP:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface MgmtEth 0/RSP0/CPU0/0
RP/0/RSP0/CPU0:router(config)# ipv4 address 172.29.52.70 255.255.255.0
RP/0/RSP0/CPU0:router(config-if)# speed 100
RP/0/RSP0/CPU0:router(config-if)# duplex full
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# commit
RP/0/RSP0/CPU0:Mar 26 01:09:28.685 :ifmgr[190]:%LINK-3-UPDOWN :Interface
MgmtEth0/RSP0/CPU0/0, changed state to Up
RP/0/RSP0/CPU0:router(config-if)# end

RP/0/RSP0/CPU0:router# show interfaces MgmtEth 0/RSP0/CPU0/0

MMgmtEth0/RSP0/CPU0/0 is up, line protocol is up
  Hardware is Management Ethernet, address is 0011.93ef.e8ea (bia 0011.93ef.e8ea
)
  Description: Connected to Lab LAN
  Internet address is 172.29.52.70/24
  MTU 1514 bytes, BW 100000 Kbit
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA,  loopback not set,
  ARP type ARPA, ARP timeout 04:00:00
  Last clearing of "show interface" counters never
  5 minute input rate 3000 bits/sec, 7 packets/sec
  5 minute output rate 0 bits/sec, 1 packets/sec
     30445 packets input, 1839328 bytes, 64 total input drops
     0 drops for unrecognized upper-level protocol
     Received 23564 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     57 input errors, 40 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     171672 packets output, 8029024 bytes, 0 total output drops
     Output 16 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     1 carrier transitions

RP/0/RSP0/CPU0:router# show running-config interface MgmtEth 0/RSP0/CPU0/0

interface MgmtEth0/RSP0/CPU0/0
 description Connected to Lab LAN
 ipv4 address 172.29.52.70 255.255.255.0
!
```

Additional References

The following sections provide references related to Management Ethernet interface configuration.

Related Documents

Related Topic	Document Title
Cisco ASR 9000 Series Router master command reference	*Cisco ASR 9000 Series Router Master Commands List*
Cisco ASR 9000 Series Router interface configuration commands	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a Cisco ASR 9000 Series Router using the Cisco IOS XR Software.	*Cisco ASR 9000 Series Router Getting Started Guide*
Information about user groups and task IDs	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by the feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Ethernet Interfaces on the Cisco ASR 9000 Series Router

This module describes the configuration of Ethernet interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

The distributed Gigabit Ethernet and 10-Gigabit Ethernet architecture and features deliver network scalability and performance, while enabling service providers to offer high-density, high-bandwidth networking solutions designed to interconnect the router with other systems in POPs, including core and edge routers and Layer 2 and Layer 3 switches.

Feature History for Configuring Ethernet Interfaces on the Cisco ASR 9000 Series Router

Release	Modification
Release 3.7.2	Support was added on the Cisco ASR 9000 Series Router for the following line cards: • 40-Port Gigabit Ethernet Medium Queue and High Queue Line Cards (A9K-40GE-B and A9K-40GE-E) • 4-Port 10-Gigabit Ethernet Medium Queue and High Queue Line Cards (A9K-4T-B and A9K-4T-E) • 8-Port 10-Gigabit Ethernet Medium Queue and High Queue DX Line Cards (A9K-8T/4-B and A9K-8T/4-E) (2:1 oversubscribed)

Release 3.9.0	Support was added on the Cisco ASR 9000 Series Router for the following line cards: • 40-Port Gigabit Ethernet Low Queue Line Card (A9K-40GE-L) • 4-Port 10-Gigabit Ethernet Low Queue Line Card (A9K-4T-L) • 8-Port 10-Gigabit Ethernet Low Queue DX Line Card (A9K-8T/4-L) (2:1 oversubscribed) • 8-Port 10-Gigabit Ethernet Low and High Queue Line Card (A9K-8T-L and A9K-8T-E) • 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Medium Queue and High Queue Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L) Support for the following features was added: • Frequency Synchronization • SyncE
Release 3.9.1	Support was added on the Cisco ASR 9000 Series Router for the following line cards: • 8-Port 10-Gigabit Ethernet Medium Queue Line Card (A9K-8T-B) • 16-Port 10-Gigabit Ethernet SFP+ Line Card (A9K-16T/8-B and A9K-16T/8-B+AIP)
Release 4.0.1	Support for Layer 2 statistics collection for performance monitoring on Layer 2 subinterfaces (EFPs) is added.
Release 4.1.0	Support for Link Layer Discovery Protocol (LLDP) was added.
Release 4.1.1	Support was added for MAC address accounting feature.
Release 4.2.3	Support for Autonegotiation feature on Gigabit Ethernet interfaces was added.

Contents

Prerequisites for Configuring Ethernet Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring Ethernet interfaces, be sure that the following tasks and conditions are met:

- Confirm that at least one of the following line cards supported on the router is installed:
  - 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination line card (A9K-2T20GE-B and A9K-2T20GE-L)
  - 4-Port 10-Gigabit Ethernet line card (A9K-4T-L, -B, or -E)
  - 8-Port 10-Gigabit Ethernet DX line card (A9K-8T/4-L, -B, or -E)
  - 8-Port 10-Gigabit Ethernet line card (A9K-8T-L, -B, or -E)
  - 16-Port 10-Gigabit Ethernet SFP+ line card (A9K-16T/8-B and A9K-16T/8-B+AIP)
  - 40-Port Gigabit Ethernet line card (A9K-40GE-L, -B, or -E)
- Know the interface IP address.
- You know how to apply the specify the generalized interface name with the generalized notation *rack/slot/module/port*.

Information About Configuring Ethernet

Ethernet is defined by the IEEE 802.3 international standard. It enables the connection of up to 1024 nodes over coaxial, twisted-pair, or fiber-optic cable.

The Cisco ASR 9000 Series Router supports Gigabit Ethernet (1000 Mbps) and 10-Gigabit Ethernet (10 Gbps) interfaces.

This section provides the following information sections:

16-Port 10-Gigabit Ethernet SFP+ Line Card

The 16-Port10-Gigabit Ethernet SFP+ line card is a Small Form Factor (SFP transceiver) optical line card introduced in Cisco IOS XR Release 3.9.1 on the Cisco ASR 9000 Series Router. The 16-Port10-Gigabit Ethernet SFP+ line card supports all of the Gigabit Ethernet commands and configurations currently supported on the router.

The 16-Port10-Gigabit Ethernet SFP+ line card is compatible with all existing Cisco ASR 9000 Series Router line cards, route/switch processors (RSPs), and chassis.

Features

The 16-Port10-Gigabit Ethernet SFP+ line card supports the following features:

- 16 10-Gigabit Ethernet ports
- 128 10-Gigabit Ethernet ports per system
- 1.28 Tbps per system
- 160 Gbps forwarding
- 120 Gbps bidirectional performance
- SR/LR/ER SFP+ optics

- Feature parity with existing line cards
- Unicast and multicast forwarding at 160 Gbps, with zero packet loss during RSP switchover

Restrictions

The following features are not supported on the 16-Port10-Gigabit Ethernet SFP+ line card:

- DWDM (G.709)

Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet

Table 3 describes the default interface configuration parameters that are present when an interface is enabled on a Gigabit Ethernet or 10-Gigabit Ethernet modular services card and its associated PLIM.

Note You must use the **shutdown** command to bring an interface administratively down. The interface default is **no shutdown**. When a modular services card is first inserted into the router, if there is no established preconfiguration for it, the configuration manager adds a shutdown item to its configuration. This shutdown can be removed only be entering the **no shutdown** command.

Table 3 Gigabit Ethernet and 10-Gigabit Ethernet Modular Services Card Default Configuration Values

Parameter	Configuration File Entry	Default Value
MAC accounting	**mac-accounting**	**off**
Flow control	**flow-control**	egress on ingress off
MTU	**mtu**	• 1514 bytes for normal frames • 1518 bytes for 802.1Q tagged frames. • 1522 bytes for Q-in-Q frames.
MAC address	**mac address**	Hardware burned-in address (BIA)

Table 4 Fast Ethernet Default Configuration Values

Parameter	Configuration File Entry	Default Value
MAC accounting	**mac-accounting**	**off**
Duplex operation	**duplex full** **duplex half**	Auto-negotiates duplex operation
MTU	**mtu**	1500 bytes

Table 4 Fast Ethernet Default Configuration Values

Parameter	Configuration File Entry	Default Value
Interface speed	**speed**	100 Mbps
Auto-negotiation	**negotiation auto**	**disable**

Layer 2 VPN on Ethernet Interfaces

Layer 2 Virtual Private Network (L2VPN) connections emulate the behavior of a LAN across an L2 switched, IP or MPLS-enabled IP network, allowing Ethernet devices to communicate with each other as if they were connected to a common LAN segment.

The L2VPN feature enables service providers (SPs) to provide Layer 2 services to geographically disparate customer sites. Typically, an SP uses an access network to connect the customer to the core network. On the Cisco ASR 9000 Series Router, this access network is typically Ethernet.

Traffic from the customer travels over this link to the edge of the SP core network. The traffic then tunnels through an L2VPN over the SP core network to another edge router. The edge router sends the traffic down another attachment circuit (AC) to the customer's remote site.

On the Cisco ASR 9000 Series Router, an AC is an interface that is attached to an L2VPN component, such as a bridge domain, pseudowire, or local connect.

The L2VPN feature enables users to implement different types of end-to-end services.

Cisco IOS XR software supports a point-to-point end-to-end service, where two Ethernet circuits are connected together. An L2VPN Ethernet port can operate in one of two modes:

- Port Mode—In this mode, all packets reaching the port are sent over the PW (pseudowire), regardless of any VLAN tags that are present on the packets. In VLAN mode, the configuration is performed under the l2transport configuration mode.
- VLAN Mode—Each VLAN on a CE (customer edge) or access network to PE (provider edge) link can be configured as a separate L2VPN connection (using either VC type 4 or VC type 5). In VLAN mode, the configuration is performed under the individual subinterface.

Switching can take place in three ways:

- AC-to-PW—Traffic reaching the PE is tunneled over a PW (and conversely, traffic arriving over the PW is sent out over the AC). This is the most common scenario.
- Local switching—Traffic arriving on one AC is immediately sent out of another AC without passing through a pseudowire.
- PW stitching—Traffic arriving on a PW is not sent to an AC, but is sent back into the core over another PW.

Keep the following in mind when configuring L2VPN on an Ethernet interface:

- L2VPN links support QoS (Quality of Service) and MTU (maximum transmission unit) configuration.
- If your network requires that packets are transported transparently, you may need to modify the packet's destination MAC (Media Access Control) address at the edge of the Service Provider (SP) network. This prevents the packet from being consumed by the devices in the SP network.

Use the **show interfaces** command to display AC and PW information.

To configure a point-to-point pseudowire xconnect on an AC, refer to these documents:

- *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*
- *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*

To attach Layer 2 service policies, such as QoS, to the Ethernet interface, refer to the appropriate Cisco IOS XR software configuration guide.

Gigabit Ethernet Protocol Standards Overview

The Gigabit Ethernet interfaces support the following protocol standards:

- IEEE 802.3 Physical Ethernet Infrastructure, page 27
- IEEE 802.3ab 1000BASE-T Gigabit Ethernet, page 27
- IEEE 802.3z 1000 Mbps Gigabit Ethernet, page 27
- IEEE 802.3ae 10 Gbps Ethernet, page 28

These standards are further described in the sections that follow.

IEEE 802.3 Physical Ethernet Infrastructure

The IEEE 802.3 protocol standards define the physical layer and MAC sublayer of the data link layer of wired Ethernet. IEEE 802.3 uses Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access at a variety of speeds over a variety of physical media. The IEEE 802.3 standard covers 10 Mbps Ethernet. Extensions to the IEEE 802.3 standard specify implementations for Gigabit Ethernet, 10-Gigabit Ethernet, and Fast Ethernet.

IEEE 802.3ab 1000BASE-T Gigabit Ethernet

The IEEE 802.3ab protocol standards, or Gigabit Ethernet over copper (also known as 1000BaseT) is an extension of the existing Fast Ethernet standard. It specifies Gigabit Ethernet operation over the Category 5e/6 cabling systems already installed, making it a highly cost-effective solution. As a result, most copper-based environments that run Fast Ethernet can also run Gigabit Ethernet over the existing network infrastructure to dramatically boost network performance for demanding applications.

IEEE 802.3z 1000 Mbps Gigabit Ethernet

Gigabit Ethernet builds on top of the Ethernet protocol, but increases speed tenfold over Fast Ethernet to 1000 Mbps, or 1 Gbps. Gigabit Ethernet allows Ethernet to scale from 10 or 100 Mbps at the desktop to 100 Mbps up to 1000 Mbps in the data center. Gigabit Ethernet conforms to the IEEE 802.3z protocol standard.

By leveraging the current Ethernet standard and the installed base of Ethernet and Fast Ethernet switches and routers, network managers do not need to retrain and relearn a new technology in order to provide support for Gigabit Ethernet.

IEEE 802.3ae 10 Gbps Ethernet

Under the International Standards Organization's Open Systems Interconnection (OSI) model, Ethernet is fundamentally a Layer 2 protocol. 10-Gigabit Ethernet uses the IEEE 802.3 Ethernet MAC protocol, the IEEE 802.3 Ethernet frame format, and the minimum and maximum IEEE 802.3 frame size. 10 Gbps Ethernet conforms to the IEEE 802.3ae protocol standards.

Just as 1000BASE-X and 1000BASE-T (Gigabit Ethernet) remained true to the Ethernet model, 10-Gigabit Ethernet continues the natural evolution of Ethernet in speed and distance. Because it is a full-duplex only and fiber-only technology, it does not need the carrier-sensing multiple-access with the CSMA/CD protocol that defines slower, half-duplex Ethernet technologies. In every other respect, 10-Gigabit Ethernet remains true to the original Ethernet model.

IEEE 802.3ba 100 Gbps Ethernet

IEEE 802.3ba is supported on the Cisco 1-Port 100-Gigabit Ethernet PLIM beginning in Cisco IOS XR 4.0.1.

MAC Address

A MAC address is a unique 6-byte address that identifies the interface at Layer 2.

MAC Accounting

The MAC address accounting feature provides accounting information for IP traffic based on the source and destination MAC addresses on LAN interfaces. This feature calculates the total packet and byte counts for a LAN interface that receives or sends IP packets to or from a unique MAC address. It also records a time stamp for the last packet received or sent.

These statistics are used for traffic monitoring, debugging and billing. For example, with this feature you can determine the volume of traffic that is being sent to and/or received from various peers at NAPS/peering points. This feature is currently supported on Ethernet, FastEthernet, and bundle interfaces and supports Cisco Express Forwarding (CEF), distributed CEF (dCEF), flow, and optimum switching.

Note A maximum of 512 MAC addresses per trunk interface are supported for MAC address accounting.

Ethernet MTU

The Ethernet maximum transmission unit (MTU) is the size of the largest frame, minus the 4-byte frame check sequence (FCS), that can be transmitted on the Ethernet network. Every physical network along the destination of a packet can have a different MTU.

Cisco IOS XR software supports two types of frame forwarding processes:

- Fragmentation for IPV4 packets–In this process, IPv4 packets are fragmented as necessary to fit within the MTU of the next-hop physical network.

Note IPv6 does not support fragmentation.

- MTU discovery process determines largest packet size–This process is available for all IPV6 devices, and for originating IPv4 devices. In this process, the originating IP device determines the size of the largest IPv6 or IPV4 packet that can be sent without being fragmented. The largest packet is equal to the smallest MTU of any network between the IP source and the IP destination devices. If a packet is larger than the smallest MTU of all the networks in its path, that packet will be fragmented as necessary. This process ensures that the originating device does not send an IP packet that is too large.

Jumbo frame support is automatically enable for frames that exceed the standard frame size. The default value is 1514 for standard frames and 1518 for 802.1Q tagged frames. These numbers exclude the 4-byte frame check sequence (FCS).

Flow Control on Ethernet Interfaces

The flow control used on 10-Gigabit Ethernet interfaces consists of periodically sending flow control pause frames. It is fundamentally different from the usual full- and half-duplex flow control used on standard management interfaces. Flow control can be activated or deactivated for ingress traffic only. It is automatically implemented for egress traffic.

802.1Q VLAN

A VLAN is a group of devices on one or more LANs that are configured so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments. Because VLANs are based on logical instead of physical connections, it is very flexible for user and host management, bandwidth allocation, and resource optimization.

The IEEE's 802.1Q protocol standard addresses the problem of breaking large networks into smaller parts so broadcast and multicast traffic does not consume more bandwidth than necessary. The standard also helps provide a higher level of security between segments of internal networks.

The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames.

VRRP

The Virtual Router Redundancy Protocol (VRRP) eliminates the single point of failure inherent in the static default routed environment. VRRP specifies an election protocol that dynamically assigns responsibility for a virtual router to one of the VPN concentrators on a LAN. The VRRP VPN concentrator controlling the IP addresses associated with a virtual router is called the master, and forwards packets sent to those IP addresses. When the master becomes unavailable, a backup VPN concentrator takes the place of the master.

For more information on VRRP, see the *Implementing VRRP* module of *Cisco ASR 9000 Series Router IP Addresses and Services Configuration Guide.*

HSRP

Hot Standby Routing Protocol (HSRP) is a proprietary protocol from Cisco. HSRP is a routing protocol that provides backup to a router in the event of failure. Several routers are connected to the same segment of an Ethernet, FDDI, or token-ring network and work together to present the appearance of a single virtual router on the LAN. The routers share the same IP and MAC addresses and therefore, in the event of failure of one router, the hosts on the LAN are able to continue forwarding packets to a consistent IP and MAC address. The transfer of routing responsibilities from one device to another is transparent to the user.

HSRP is designed to support non disruptive switchover of IP traffic in certain circumstances and to allow hosts to appear to use a single router and to maintain connectivity even if the actual first hop router they are using fails. In other words, HSRP protects against the failure of the first hop router when the source host cannot learn the IP address of the first hop router dynamically. Multiple routers participate in HSRP and in concert create the illusion of a single virtual router. HSRP ensures that one and only one of the routers is forwarding packets on behalf of the virtual router. End hosts forward their packets to the virtual router.

The router forwarding packets is known as the *active router*. A standby router is selected to replace the active router should it fail. HSRP provides a mechanism for determining active and standby routers, using the IP addresses on the participating routers. If an active router fails a standby router can take over without a major interruption in the host's connectivity.

HSRP runs on top of User Datagram Protocol (UDP), and uses port number 1985. Routers use their actual IP address as the source address for protocol packets, not the virtual IP address, so that the HSRP routers can identify each other.

For more information on HSRP, see the *Implementing HSRP* module of *Cisco ASR 9000 Series Router IP Addresses and Services Configuration Guide.*

Link Autonegotiation on Ethernet Interfaces

Link autonegotiation ensures that devices that share a link segment are automatically configured with the highest performance mode of interoperation. Use the **negotiation auto** command in interface configuration mode to enable link autonegotiation on an Ethernet interface. On line card Ethernet interfaces, link autonegotiation is disabled by default.

Note The **negotiation auto** command is available on Gigabit Ethernet interfaces only.

Table 5 describes the performance of the system for different combinations of the speed modes. The specified **speed** command produces the resulting system action, provided that you have configured autonegotiation on the interface.

Table 5 Relationship Between duplex and speed Commands

duplex Command	speed Command	Resulting System Action
no duplex	**no speed**	Auto-negotiates both speed and duplex modes.
no duplex	**speed 1000**	Auto-negotiates for duplex mode and forces 1000 Mbps.
no duplex	**speed 100**	Auto-negotiates for duplex mode and forces 100 Mbps.

Table 5 Relationship Between duplex and speed Commands (continued)

duplex Command	speed Command	Resulting System Action
no duplex	**speed 10**	Auto-negotiates for duplex mode and forces 10 Mbps.
full-duplex	**no speed**	Forces full duplex and auto-negotiates for speed.
full-duplex	**speed 1000**	Forces full duplex and 1000 Mbps.
full-duplex	**speed 100**	Forces full duplex and 100 Mbps.
full-duplex	**speed 10**	Forces full duplex and 10 Mbps.
half-duplex	**no speed**	Forces half duplex and auto-negotiates for speed.
half-duplex	**speed 1000**	Forces half duplex and 1000 Mbps.
half-duplex	**speed 100**	Forces half duplex and 100 Mbps.
half-duplex	**speed 10**	Forces half duplex and 10 Mbps.

Subinterfaces on the Cisco ASR 9000 Series Router

In Cisco IOS XR, interfaces are main interfaces by default. A main interface is also called a trunk interface, which is not to be confused with the usage of the word trunk in the context of VLAN trunking.

There are three types of trunk interfaces:

- Physical
- Bundle

On the Cisco ASR 9000 Series Router, physical interfaces are automatically created when the router recognizes a card and its physical interfaces. However, bundle interfaces are not automatically created. They are created when they are configured by the user.

The following configuration samples are examples of trunk interfaces being created:

- interface gigabitethernet 0/5/0/0
- interface bundle-ether 1

A subinterface is a logical interface that is created under a trunk interface.

To create a subinterface, the user must first identify a trunk interface under which to place it. In the case of bundle interfaces, if one does not already exist, a bundle interface must be created before any subinterfaces can be created under it.

The user then assigns a subinterface number to the subinterface to be created. The subinterface number must be a positive integer from zero to some high value. For a given trunk interface, each subinterface under it must have a unique value.

Subinterface numbers do not need to be contiguous or in numeric order. For example, the following subinterfaces numbers would be valid under one trunk interface:

1001, 0, 97, 96, 100000

Subinterfaces can never have the same subinterface number under one trunk.

In the following example, the card in slot 5 has trunk interface, GigabitEthernet 0/5/0/0. A subinterface, GigabitEthernet 0/5/0/0.0, is created under it.

```
RP/0/RSP0/CPU0:router#  conf
Mon Sep 21 11:12:11.722 EDT
RP/0/RSP0/CPU0:router(config)#  interface GigabitEthernet0/5/0/0.0
```

```
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-subif)# commit

RP/0/RSP0/CPU0:Sep 21 11:12:34.819 : config[65794]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'root'. Use 'show configuration commit changes
1000000152' to view the changes.

RP/0/RSP0/CPU0:router(config-subif)# end

RP/0/RSP0/CPU0:Sep 21 11:12:35.633 : config[65794]: %MGBL-SYS-5-CONFIG_I : Configured
from console by root
RP/0/RSP0/CPU0:router#
```

The **show run** command displays the trunk interface first, then the subinterfaces in ascending numerical order.

```
RP/0/RSP0/CPU0:router# show run | begin GigabitEthernet0/5/0/0
Mon Sep 21 11:15:42.654 EDT
Building configuration...
interface GigabitEthernet0/5/0/0
 shutdown
!
interface GigabitEthernet0/5/0/0.0
 encapsulation dot1q 100
!
interface GigabitEthernet0/5/0/1
 shutdown
!
```

When a subinterface is first created, the Cisco ASR 9000 Series Router recognizes it as an interface that, with few exceptions, is interchangeable with a trunk interface. After the new subinterface is configured further, the **show interface** command can display it along with its unique counters:

The following example shows the display output for the trunk interface, GigabitEthernet 0/5/0/0, followed by the display output for the subinterface GigabitEthernet 0/5/0/0.0.

```
RP/0/RSP0/CPU0:router# show interface gigabitEthernet 0/5/0/0
Mon Sep 21 11:12:51.068 EDT
GigabitEthernet0/5/0/0 is administratively down, line protocol is administratively
down
  Interface state transitions: 0
  Hardware is GigabitEthernet, address is 0024.f71b.0ca8 (bia 0024.f71b.0ca8)
  Internet address is Unknown
  MTU 1514 bytes, BW 1000000 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation 802.1Q Virtual LAN,
  Full-duplex, 1000Mb/s, SXFD, link type is force-up
  output flow control is off, input flow control is off
  loopback not set,
  ARP type ARPA, ARP timeout 04:00:00
  Last input never, output never
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions


RP/0/RSP0/CPU0:router# show interface gigabitEthernet0/5/0/0.0
Mon Sep 21 11:12:55.657 EDT
GigabitEthernet0/5/0/0.0 is administratively down, line protocol is administratively
down
  Interface state transitions: 0
  Hardware is VLAN sub-interface(s), address is 0024.f71b.0ca8
  Internet address is Unknown
  MTU 1518 bytes, BW 1000000 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation 802.1Q Virtual LAN, VLAN Id 100,  loopback not set,
  ARP type ARPA, ARP timeout 04:00:00
  Last input never, output never
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
```

The following example shows two interfaces being created at the same time: first, the bundle trunk interface, then a subinterface attached to the trunk:

```
RP/0/RSP0/CPU0:router# conf
Mon Sep 21 10:57:31.736 EDT
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether1
RP/0/RSP0/CPU0:router(config-if)# no shut
RP/0/RSP0/CPU0:router(config-if)# interface bundle-Ether1.0
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-subif)# commit
RP/0/RSP0/CPU0:Sep 21 10:58:15.305 : config[65794]: %MGBL-CONFIG-6-DB_COMMIT : C
onfiguration committed by user 'root'. Use 'show configuration commit changes 10
00000149' to view the changes.
RP/0/RSP0/CPU0:router# show run | begin Bundle-Ether1
Mon Sep 21 10:59:31.317 EDT
Building configuration...
interface Bundle-Ether1
!
interface Bundle-Ether1.0
 encapsulation dot1q 100
!
```

You delete a subinterface using the **no interface** command.

```
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router# show run | begin GigabitEthernet0/5/0/0
Mon Sep 21 11:42:27.100 EDT
Building configuration...
interface GigabitEthernet0/5/0/0
 negotiation auto
!
interface GigabitEthernet0/5/0/0.0
 encapsulation dot1q 100
!
interface GigabitEthernet0/5/0/1
 shutdown
!
RP/0/RSP0/CPU0:router# conf
Mon Sep 21 11:42:32.374 EDT
```

```
RP/0/RSP0/CPU0:router(config)# no interface GigabitEthernet0/5/0/0.0
RP/0/RSP0/CPU0:router(config)# commit
RP/0/RSP0/CPU0:Sep 21 11:42:47.237 : config[65794]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'root'. Use 'show configuration commit changes
1000000159' to view the changes.
RP/0/RSP0/CPU0:router(config)# end
RP/0/RSP0/CPU0:Sep 21 11:42:50.278 : config[65794]: %MGBL-SYS-5-CONFIG_I : Configured
from console by root
RP/0/RSP0/CPU0:router# show run | begin GigabitEthernet0/5/0/0
Mon Sep 21 11:42:57.262 EDT
Building configuration...
interface GigabitEthernet0/5/0/0
 negotiation auto
!
interface GigabitEthernet0/5/0/1
 shutdown
!
```

Layer 2, Layer 3, and EFP's

On the Cisco ASR 9000 Series Router, a trunk interface can be either a Layer 2 or Layer 3 interface. A Layer 2 interface is configured using the **interface** command with the **l2transport** keyword. When the **l2transport** keyword is not used, the interface is a Layer 3 interface. Subinterfaces are configured as Layer 2 or Layer 3 subinterface in the same way.

A Layer 3 trunk interface or subinterface is a routed interface and can be assigned an IP address. Traffic sent on that interface is routed.

A Layer 2 trunk interface or subinterface is a switched interface and cannot be assigned an IP address. A Layer 2 interface must be connected to an L2VPN component. Once it is connected, it is called an access connection.

Subinterfaces can only be created under a Layer 3 trunk interface. Subinterfaces cannot be created under a Layer 2 trunk interface.

A Layer 3 trunk interface can have any combination of Layer 2 and Layer 3 interfaces.

The following example shows an attempt to configure a subinterface under an Layer 2 trunk and the commit errors that occur. It also shows an attempt to change the Layer 2 trunk interface to an Layer 3 interface and the errors that occur because the interface already had an IP address assigned to it.

```
RP/0/RSP0/CPU0:router# config
Mon Sep 21 12:05:33.142 EDT
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/5/0/0
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.0.0.1/24
RP/0/RSP0/CPU0:router(config-if)# commit
RP/0/RSP0/CPU0:Sep 21 12:05:57.824 : config[65794]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'root'. Use 'show configuration commit changes
1000000160' to view the changes.
RP/0/RSP0/CPU0:router(config-if)# end
RP/0/RSP0/CPU0:Sep 21 12:06:01.890 : config[65794]: %MGBL-SYS-5-CONFIG_I : Configured
from console by root
RP/0/RSP0/CPU0:router# show run | begin GigabitEthernet0/5/0/0
Mon Sep 21 12:06:19.535 EDT
Building configuration...
interface GigabitEthernet0/5/0/0
 ipv4 address 10.0.0.1 255.255.255.0
 negotiation auto
!
interface GigabitEthernet0/5/0/1
 shutdown
!
```

```
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router# conf
Mon Sep 21 12:08:07.426 EDT
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/5/0/0 l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# commit

% Failed to commit one or more configuration items during a pseudo-atomic operation.
All changes made have been reverted. Please issue 'show configuration failed' from
this session to view the errors
RP/0/RSP0/CPU0:router(config-if-l2)# no ipv4 address
RP/0/RSP0/CPU0:router(config-if)# commit
RP/0/RSP0/CPU0:Sep 21 12:08:33.686 : config[65794]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'root'. Use 'show configuration commit changes
1000000161' to view the changes.
RP/0/RSP0/CPU0:router(config-if)# end
RP/0/RSP0/CPU0:Sep 21 12:08:38.726 : config[65794]: %MGBL-SYS-5-CONFIG_I : Configured
from console by root
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router# show run interface GigabitEthernet0/5/0/0
Mon Sep 21 12:09:02.471 EDT
interface GigabitEthernet0/5/0/0
 negotiation auto
 l2transport
 !
!
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router# conf
Mon Sep 21 12:09:08.658 EDT
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/5/0/0.0
                                                  ^
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/5/0/0.0
RP/0/RSP0/CPU0:router(config-subif)# commit

% Failed to commit one or more configuration items during a pseudo-atomic operation.
All changes made have been reverted. Please issue 'show configuration failed' from
this session to view the errors
RP/0/RSP0/CPU0:router(config-subif)#
RP/0/RSP0/CPU0:router(config-subif)# interface GigabitEthernet0/5/0/0
RP/0/RSP0/CPU0:router(config-if)# no l2transport
RP/0/RSP0/CPU0:router(config-if)# interface GigabitEthernet0/5/0/0.0
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 99
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 11.0.0.1/24
RP/0/RSP0/CPU0:router(config-subif)# interface GigabitEthernet0/5/0/0.1 l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 700
RP/0/RSP0/CPU0:router(config-subif)# commit
RP/0/RSP0/CPU0:Sep 21 12:11:45.896 : config[65794]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'root'. Use 'show configuration commit changes
1000000162' to view the changes.
RP/0/RSP0/CPU0:router(config-subif)# end
RP/0/RSP0/CPU0:Sep 21 12:11:50.133 : config[65794]: %MGBL-SYS-5-CONFIG_I : Configured
from console by root
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router# show run | b GigabitEthernet0/5/0/0
Mon Sep 21 12:12:00.248 EDT
Building configuration...
interface GigabitEthernet0/5/0/0
 negotiation auto
!
interface GigabitEthernet0/5/0/0.0
 ipv4 address 11.0.0.1 255.255.255.0
 encapsulation dot1q 99
!
interface GigabitEthernet0/5/0/0.1 l2transport
```

```
 encapsulation dot1q 700
!
interface GigabitEthernet0/5/0/1
 shutdown
!
```

All subinterfaces must have unique encapsulation statements, so that the router can send incoming packets and frames to the correct subinterface. If a subinterface does not have an encapsulation statement, the router will not send any traffic to it.

In Cisco IOS XR, an Ethernet Flow Point (EFP) is implemented as a Layer 2 subinterface, and consequently, a Layer 2 subinterface is often called an EFP. For more information about EFPs, see the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*.

A Layer 2 trunk interface can be used as an access connection. However, a Layer 2 trunk interface is not an EFP because an EFP, by definition, is a substream of an overall stream of traffic.

Cisco IOS XR also has other restrictions on what can be configured as a Layer 2 or Layer 3 interface. Certain configuration blocks only accept Layer 3 and not Layer 2. For example, OSPF only accepts Layer 3 trunks and subinterface. Refer to the appropriate Cisco IOS XR configuration guide for other restrictions.

Enhanced Performance Monitoring for Layer 2 Subinterfaces (EFPs)

Beginning in Cisco IOS XR Release 4.0.1, the Cisco ASR 9000 Series Router adds support for basic counters for performance monitoring on Layer 2 subinterfaces.

This section provides a summary of the new support for Layer 2 interface counters. For information about how to configure Performance Monitoring, see the "Implementing Performance Management" chapter of the *Cisco ASR 9000 Series Aggregation Services Router System Monitoring Configuration Guide*.

The **interface basic-counters** keyword has been added to support a new entity for performance statistics collection and display on Layer 2 interfaces in the following commands:

- **performance-mgmt statistics interface basic-counters**
- **performance-mgmt threshold interface basic-counters**
- **performance-mgmt apply statistics interface basic-counters**
- **performance-mgmt apply threshold interface basic-counters**
- **performance-mgmt apply monitor interface basic-counters**
- **show performance-mgmt monitor interface basic-counters**
- **show performance-mgmt statistics interface basic-counters**

The **performance-mgmt threshold interface basic-counters** command supports the following attribute values for Layer 2 statistics, which also appear in the **show performance-mgmt statistics interface basic-counters** and **show performance-mgmt monitor interface basic-counters** command:

Attribute	Description
InOctets	Bytes received (64-bit)
InPackets	Packets received (64-bit)
InputQueueDrops	Input queue drops (64-bit)
InputTotalDrops	Inbound correct packets discarded (64-bit)

Attribute	Description
InputTotalErrors	Inbound incorrect packets discarded (64-bit)
OutOctets	Bytes sent (64-bit)
OutPackets	Packets sent (64-bit)
OutputQueueDrops	Output queue drops (64-bit)
OutputTotalDrops	Outband correct packets discarded (64-bit)
OutputTotalErrors	Outband incorrect packets discarded (64-bit)

Other Performance Management Enhancements

The following additional performance management enhancements are included in Cisco IOS XR Release 4.0.1:

- You can retain performance management history statistics across a process restart or route processor (RP) failover using the new **history-persistent** keyword option for the **performance-mgmt statistics interface** command.
- You can save performance management statistics to a local file using the **performance-mgmt resources dump local** command.
- You can filter performance management instances by defining a regular expression group (**performance-mgmt regular-expression** command), which includes multiple regular expression indices that specify strings to match. You apply a defined regular expression group to one or more statistics or threshold templates in the **performance-mgmt statistics interface** or **performance-mgmt thresholds** interface commands.

Frequency Synchronization and SyncE

Cisco IOS XR Release 3.9 introduces support for SyncE-capable Ethernet on the Cisco ASR 9000 Series Router. Frequency Synchronization provides the ability to distribute precision clock signals around the network. Highly accurate timing signals are initially injected into the Cisco ASR 9000 router in the network from an external timing technology (such as Cesium atomic clocks, or GPS), and used to clock the router's physical interfaces. Peer routers can then recover this precision frequency from the line, and also transfer it around the network. This feature is traditionally applicable to SONET/SDH networks, but with Cisco IOS XR Release 3.9, is now provided over Ethernet for Cisco ASR 9000 Series Aggregation Services Routers with Synchronous Ethernet capability.

```
interface <intf>
  <frequency synchronization config>

controller <sonet controller>
  <frequency synchronization config>

clock-interface sync <port-num> location <node>
  <additional PD commands>
  <frequency synchronization config>
```

Where `<frequency synchronization config>` expands to:

```
  frequency synchronization
    selection input
    ssm disable
    priority <pri>
    quality transmit { lowest <ql option> <ql> [ highest <ql> ] |
                       highest <ql option> <ql> |
                       exact <ql option> <ql> }
    quality receive { lowest <ql option> <ql> [ highest <ql> ] |
                      highest <ql option> <ql> |
                      exact <ql option> <ql> }
    wait-to-restore <time>
    <additional PD commands>
```

Where:

```
    <ql option> = itu-t option { 1 | 2 generation { 1 | 2 } }
frequency synchronization
  clock-interface { independent | system }
  quality itu-t option { 1 | 2 generation { 1 | 2 }}
  log selection { changes | errors }
  <additional PD commands>
```

Synchronous Ethernet is the ability to provide PHY-level frequency distribution through an Ethernet port. Previously, SDH and SONET devices were used in conjunction with external timing technology (primary reference clock [PRC] or primary reference source [PRS] using Cesium oscillators and / or global positioning system [GPS] as the clock source) to provide accurate and stable frequency reference. Using similar external references as a source, SyncE, natively supported on the Cisco ASR 9000 Series Routers, aims to achieve the same function.

Link Layer Discovery Protocol (LLDP)

The Cisco Discovery Protocol (CDP) is a device discovery protocol that runs over Layer 2 (the Data Link layer) on all Cisco-manufactured devices (routers, bridges, access servers, and switches). CDP allows network management applications to automatically discover, and acquire knowledge about, other Cisco devices connected to the network.

To support non-Cisco devices, and to allow for interoperability between other devices, the Cisco ASR 9000 Series Router also supports the IEEE 802.1AB Link Layer Discovery Protocol (LLDP). LLDP is a neighbor discovery protocol that is used by network devices to advertise information about themselves, to other devices on the network. This protocol runs over the Data Link Layer, which permits two systems, running different network layer protocols, to learn about each other.

LLDP supports a set of attributes that it uses to learn information about neighbor devices. These attributes have a defined format that is known as a Type-Length-Value (TLV). LLDP supported devices can use TLVs to receive and send information to their neighbors. Details such as configuration information, device capabilities, and device identities can be advertised using this protocol.

In addition to mandatory TLVs (Chassis ID, Port ID, and Time-to-Live), the router also supports these basic management TLVs that are optional:

- Port Description
- System Name
- System Description
- System Capabilities
- Management Address

These optional TLVs are automatically sent to the neighboring devices when LLDP is active, but you can choose to disable them, using the **lldp tlv-select disable** command.

LLDP Frame Format

LLDP frames use the IEEE 802.3 format, which consists of these fields:

- Destination address (6 bytes)—Uses a multicast address of 01-80-C2-00-00-0E.
- Source address (6 bytes)—MAC address of the sending device or port.
- LLDP Ethertype (2 bytes)—Uses 88-CC.
- LLDP PDU (1500 bytes)—LLDP payload consisting of TLVs.
- FCS (4 bytes)—Cyclic Redundancy Check (CRC) for error checking.

LLDP TLV Format

LLDP TLVs carry the information about neighboring devices within the LLDP PDU using these basic formats:

- TLV Header (16 bits), which includes these fields:
  - TLV Type (7 bits)
  - TLV Information String Length (9 bits)
- TLV Information String (0 to 511 bytes)

LLDP Operation

LLDP is a one-way protocol. The basic operation of LLDP consists of a sending device, which is enabled for transmitting LLDP information, and which sends periodic advertisements of information in LLDP frames to a receiving device.

Devices are identified using a combination of Chassis ID and Port ID TLVs to create an MSAP (MAC Service Access Point). The receiving device saves the information about a neighbor for a certain amount of time specified in the TTL TLV, before aging and removing the information.

LLDP supports these additional operational characteristics:

- LLDP operates independently in transmit or receive modes.
- LLDP operates as a slow protocol using only untagged frames, with transmission speeds of less than 5 frames per second.
- LLDP packets are sent when these events occur:
  - The packet update frequency, specified by the **lldp timer** command, is reached. The default is 30.
  - A change in the values of the managed objects occurs from the local system’s LLDP MIB.
  - LLDP is activated on an interface (3 frames are sent upon activation similar to CDP).
- When an LLDP frame is received, the LLDP remote services and PTOPO MIBs are updated with the information in the TLVs.

LLDP supports these actions on these TLV characteristics:

  - Interprets a TTL value of 0 as a request to automatically purge the information about the transmitting device. These shutdown LLDPDUs are typically sent prior to a port becoming inoperable.
  - An LLDP frame with a malformed mandatory TLV is dropped.
  - A TLV with an invalid value is ignored.

  - If the TTL is non-zero, copy of an unknown organizationally-specific TLV is maintained, for later access through network management.

Supported LLDP Functions

The Cisco ASR 9000 Series Router supports these LLDP functions:

- IPv4 and IPv6 management addresses—In general, both IPv4 and IPv6 addresses are advertised if they are available, and preference is given to the address that is configured on the transmitting interface.

  If the transmitting interface does not have a configured address, then the TLV is populated with an address from another interface. The advertised LLDP IP address is implemented according to this priority order of IP addresses for interfaces on the Cisco ASR 9000 Series Router:

  - Locally configured address
  - MgmtEth0/RSP0/CPU0/0
  - MgmtEth0/RSP0/CPU0/1
  - MgmtEth0/RSP1/CPU0/0
  - MgmtEth0/RSP1/CPU0/1
  - Loopback address

Note There are certain differences between IPv4 and IPv6 address management in LLDP:

- For IPv4, as long as the IPv4 address is configured on an interface, it can be used as an LLDP management address.
- For IPv6, after the IPv6 address is configured on an interface, the interface status must be Up and pass the Duplicate Address Detection(DAD) process before it is can be used as an LLDP management address.

- LLDP is supported for the nearest physically attached, non-tunneled neighbors.
- Port ID TLVs are supported for Ethernet interfaces, subinterfaces, bundle interfaces, and bundle subinterfaces.

Unsupported LLDP Functions

These LLDP functions are not supported on the Cisco ASR 9000 Series Router:

- LLDP-MED organizationally unique extension—Interoperability, however, still exists between other devices that do support this extension.
- Tunneled neighbors, or neighbors more than one hop away.
- LLDP TLVs cannot be disabled on a per-interface basis; Certain optional TLVs, however, can be disabled globally.

How to Configure Ethernet

This section provides the following configuration procedures:

- Configuring Ethernet Interfaces, page 41
- Configuring Frequency Synchronization and SyncE, page 48
- Configuring LLDP, page 51

Configuring Ethernet Interfaces

This section provides the following configuration procedures:

- Configuring Gigabit Ethernet Interfaces, page 41
- Configuring a L2VPN Ethernet Port, page 46
- Configuring MAC Accounting on an Ethernet Interface, page 44

Configuring Gigabit Ethernet Interfaces

Use the following procedure to create a basic Gigabit Ethernet or 10-Gigabit Ethernet interface configuration.

SUMMARY STEPS

1. **show version**
2. **show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **configure**
4. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
5. **ipv4 address** *ip-address mask*
6. **flow-control** {**bidirectional** | **egress** | **ingress**}
7. **mtu** *bytes*
8. **mac-address** *value1.value2.value3*
9. **negotiation auto** (on Gigabit Ethernet interfaces only)
10. **no shutdown**
11. **end**
    or
    commit
12. **show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show version** **Example:** `RP/0/RSP0/CPU0:router# show version`	(Optional) Displays the current software version, and can also be used to confirm that the router recognizes the modular services card.
Step 2	**show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show interface TenGigE 0/1/0/0`	(Optional) Displays the configured interface and checks the status of each interface port. Possible interface types for this procedure are: • GigabitEthernet • TenGigE
Step 3	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 4	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/1/0/0`	Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. Possible interface types for this procedure are: • GigabitEthernet • TenGigE **Note** The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1.
Step 5	**ipv4 address** *ip-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224`	Assigns an IP address and subnet mask to the interface. • Replace *ip-address* with the primary IPv4 address for the interface. • Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways: – The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address. – The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

	Command or Action	Purpose

Step 6	`flow-control {bidirectional| egress | ingress}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# flow control ingress`
		(Optional) Enables the sending and processing of flow control pause frames.
		• **egress**—Enables the sending of flow control pause frames in egress.
		• **ingress**—Enables the processing of received pause frames on ingress.
		• **bidirectional**—Enables the sending of flow control pause frames in egress and the processing of received pause frames on ingress.

Step 7	`mtu` *`bytes`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# mtu 1448`
		(Optional) Sets the MTU value for the interface.
		• The default is 1514 bytes for normal frames and 1518 bytes for 802.1Q tagged frames.
		• The range for Gigabit Ethernet and 10-Gigabit Ethernet mtu values is 64 bytes to 65535 bytes.

Step 8	`mac-address` *`value1.value2.value3`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# mac address 0001.2468.ABCD`
		(Optional) Sets the MAC layer address of the Management Ethernet interface.
		• The values are the high, middle, and low 2 bytes, respectively, of the MAC address in hexadecimal. The range of each 2-byte value is 0 to ffff.

Step 9	`negotiation auto`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# negotiation auto`
		(Optional) Enables autonegotiation on a Gigabit Ethernet interface.
		• Autonegotiation must be explicitly enabled on both ends of the connection, or speed and duplex settings must be configured manually on both ends of the connection.
		• If autonegotiation is enabled, any speed or duplex settings that you configure manually take precedence.
		Note The **negotiation auto** command is available on Gigabit Ethernet interfaces only.

Step 10	`no shutdown`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration, which forces an interface administratively down.

	Command or Action	Purpose

Step 11	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 12	**show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show interfaces TenGigE 0/3/0/0`	(Optional) Displays statistics for interfaces on the router.

What to Do Next

- To attach Layer 3 service policies, such as Multiprotocol Label Switching (MPLS) or Quality of Service (QoS), to the Ethernet interface, refer to the appropriate Cisco ASR 9000 Series Router configuration guide.

Configuring MAC Accounting on an Ethernet Interface

This task explains how to configure MAC accounting on an Ethernet interface. MAC accounting has special **show** commands, which are illustrated in this procedure. Otherwise, the configuration is the same as configuring a basic Ethernet interface, and the steps can be combined in one configuration session. See “Configuring Gigabit Ethernet Interfaces” in this module for information about configuring the other common parameters for Ethernet interfaces.

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE** | **fastethernet**] *interface-path-id*
3. **ipv4 address** *ip-address mask*
4. **mac-accounting** {**egress** | **ingress**}

5. **end**
   or
   commit
6. **show mac-accounting** *type* **location** *instance*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`** `|` **`fastethernet`**`]` *`interface-path-id`*
	Example:
	`RP/0/RP0/CPU0:router(config)# interface TenGigE 0/1/0/0`
		Physical interface or virtual interface.
		Note Use the **show interfaces** command to see a list of all interfaces currently configured on the router.
		For more information about the syntax for the router, use the question mark (**?**) online help function.

Step 3	**`ipv4 address`** *`ip-address mask`*
	Example:
	`RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224`
		Assigns an IP address and subnet mask to the interface.
		• Replace *ip-address* with the primary IPv4 address for the interface.
		• Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways:
		– The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address.
		– The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

Step 4	**`mac-accounting`** `{`**`egress`** `|` **`ingress`**`}`
	Example:
	`RP/0/RP0/CPU0:router(config-if)# mac-accounting egress`
		Generates accounting information for IP traffic based on the source and destination MAC addresses on LAN interfaces.
		• To disable MAC accounting, use the **no** form of this command.

	Command or Action	Purpose

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**show mac-accounting** *type* **location** *instance* **Example:** `RP/0/RP0/CPU0:router# show mac-accounting TenGigE location 0/2/0/4`	Displays MAC accounting statistics for an interface.

Configuring a L2VPN Ethernet Port

Use the following procedure to configure an L2VPN Ethernet port.

Note The steps in this procedure configure the L2VPN Ethernet port to operate in port mode.

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **l2transport**
4. **l2protocol cpsv** {**tunnel** | **reverse-tunnel**}
5. **end**
   or
   commit
6. **show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**`configure`** **Example:** `RP/0/RSP0/CPU0:router#`	Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/1/0/0`
		Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. Possible interface types for this procedure are:
		• GigabitEthernet
		• TenGigE

Step 3	**`l2transport`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# l2transport`
		Enables Layer 2 transport mode on a port and enter Layer 2 transport configuration mode.

Step 4	**`l2protocol cpsv`** `{`**`tunnel`** `|` **`reverse-tunnel`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# l2protocol cpsv tunnel`
		Configures Layer 2 protocol tunneling and protocol data unit (PDU) filtering on an Ethernet interface for the following protocols: CDP, PVST+, STP, VTP, where:
		• **tunnel**—Specifies L2PT encapsulation on frames as they enter the interface, and de-encapsulation on frames as they exit they interface.
		• **reverse-tunnel**—Specifies L2PT encapsulation on frames as they exit the interface, and de-encapsulation on frames as they enter the interface.

	Command or Action	Purpose

Step 5	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if-l2)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**`show interfaces`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show interfaces TenGigE 0/3/0/0`
		(Optional) Displays statistics for interfaces on the router.

What to Do Next

To configure a point-to-point pseudowire xconnect on an AC, refer to these documents:

- *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*
- *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*

To attach Layer 2 service policies, such as quality of service (QoS), to the Ethernet interface, refer to the appropriate Cisco IOS XR software configuration guide.

Configuring Frequency Synchronization and SyncE

This section describes how to configure the Frequency Synchronization and SyncE feature on the Cisco ASR 9000 Series Aggregation Services Routers. It includes the following topics:

- Global Configuration, page 49
- Line Interface Configuration, page 50

Global Configuration

Use the following procedure to set up the frequency synchronization feature globally.

SUMMARY STEPS

1. **configure**
2. **frequency synchronization**
3. **end**
or
commit

Command	Purpose

`configure`
Example:
`RP/0/RSP0/CPU0:router# configure`
	Enters global configuration mode.

`frequency synchronization`
Example:
`RP/0/RSP0/CPU0:ios(config)#frequency synchronization`
	Enables frequency synchronization for all interfaces.

`end`
or
`commit`
Example:
`RP/0/RSP0/CPU0:router(config-freqsync)# end`
or
`RP/0/RSP0/CPU0:router(config-freqsync)# commit`
	Saves configuration changes.
	• When you issue the **end** command, the system prompts you to commit changes:
	`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
	– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
	– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
	– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
	• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Line Interface Configuration

Use the following procedure to create a basic Gigabit Ethernet or 10-Gigabit Ethernet interface configuration and enable the frequency synchronization feature on the configured line interface.

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **frequency synchronization**
4. **end**
   or
   commit

Command	Purpose

`configure`
Example:
`RP/0/RSP0/CPU0:router# configure`
	Enters global configuration mode.

`interface GigabitEthernet 0/2/0/0`
Example:
`RP/0/RSP0/CPU0:ios(config)##interface GigabitEthernet 0/2/0/0`
	Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. Possible interface types for this procedure are:
	• GigabitEthernet
	• TenGigE
	Note The example indicates a Gigabit Ethernet interface in modular services card slot 2.

Command	Purpose
frequency synchronization **Example:** `RP/0/RSP0/CPU0:ios(config-if)#`**`frequency synchronization`**	Enables frequency synchronization for all interfaces.
end or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-freqsync)#` **`end`** or `RP/0/RSP0/CPU0:router(config-freqsync)#` **`commit`**	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring LLDP

This section includes these configuration topics for LLDP:

- LLDP Default Configuration, page 52
- Enabling LLDP Globally, page 52 (required)
- Configuring Global LLDP Operational Characteristics, page 53 (optional)
- Disabling Transmission of Optional LLDP TLVs, page 55 (optional)
- Disabling LLDP Receive and Transmit Operations for an Interface, page 56 (optional)
- Verifying the LLDP Configuration, page 58

LLDP Default Configuration

Table 6 shows values of the LLDP default configuration on the Cisco ASR 9000 Series Router. To change the default settings, use the LLDP global configuration and LLDP interface configuration commands.

Table 6 LLDP Default Configuration

LLDP Function	Default
LLDP global state	Disabled
LLDP holdtime (before discarding), in seconds	120
LLDP timer (packet update frequency), in seconds	30
LLDP reinitialization delay, in seconds	2
LLDP TLV selection	All TLVs are enabled for sending and receiving.
LLDP interface state	Enabled for both transmit and receive operations when LLDP is globally enabled.

Enabling LLDP Globally

To run LLDP on the router, you must enable it globally. When you enable LLDP globally, all interfaces that support LLDP are automatically enabled for both transmit and receive operations.

You can override this default operation at the interface to disable receive or transmit operations. For more information about how to selectively disable LLDP receive or transmit operations for an interface, see the "Disabling LLDP Receive and Transmit Operations for an Interface" section on page 56.

To enable LLDP globally, complete these steps:

SUMMARY STEPS

1. **configure**
2. **lldp**
3. **end**
   or
   commit

	Command	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

	Command	Purpose

Step 2	`lldp`
	Example:
	`RP/0/RSP0/CPU0:router(config)# lldp`
		Enables LLDP globally for both transmit and receive operations on the system.

Step 3	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file, and remain within the configuration session.

Configuring Global LLDP Operational Characteristics

The "LLDP Default Configuration" section on page 52 describes the default operational characteristics for LLDP. When you enable LLDP globally on the router using the **lldp** command, these defaults are used for the protocol.

To modify the global LLDP operational characteristics such as the LLDP neighbor information holdtime, initialization delay, or packet rate, complete these steps:

SUMMARY STEPS

1. **configure**
2. **lldp holdtime** *seconds*
3. **lldp reinit** *seconds*
4. **lldp timer** *seconds*
5. **end**
   or
   commit

	Command	Purpose
Step 1	**configure** **Example:** RP/0/RSP0/CPU0:router# **configure**	Enters global configuration mode.
Step 2	**lldp holdtime** *seconds* **Example:** RP/0/RSP0/CPU0:router(config)# **lldp holdtime 60**	(Optional) Specifies the length of time that information from an LLDP packet should be held by the receiving device before aging and removing it.
Step 3	**lldp reinit** *seconds* **Example:** RP/0/RSP0/CPU0:router(config)# **lldp reinit 4**	(Optional) Specifies the length of time, the initialization of LLDP on an interface should be delayed.
Step 4	**lldp timer** *seconds* **Example:** RP/0/RSP0/CPU0:router(config)# **lldp reinit 60**	(Optional) Specifies the LLDP packet rate.
Step 5	**end** or **commit** **Example:** RP/0/RSP0/CPU0:router(config)# **end** or RP/0/RSP0/CPU0:router(config)# **commit**	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file, and remain within the configuration session.

Disabling Transmission of Optional LLDP TLVs

Certain TLVs are classified as mandatory in LLDP packets, such as the Chassis ID, Port ID, and Time to Live (TTL) TLVs. These TLVs must be present in every LLDP packet. You can suppress transmission of certain other optional TLVs in LLDP packets.

To disable transmission of optional LLDP TLVs, complete these steps:

SUMMARY STEPS

1. **configure**
2. **lldp tlv-select** *tlv-name* **disable**
3. **end**
   or
   commit

	Command	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router#`
	`configure`
		Enters global configuration mode.

	Command	Purpose
Step 2	**lldp tlv-select** *tlv-name* **disable** **Example:** `RP/0/RSP0/CPU0:router(config)# lldp tlv-select system-capabilities disable`	(Optional) Specifies that transmission of the selected TLV in LLDP packets is disabled. The *tlv-name* can be one of these LLDP TLV types: • **management-address** • **port-description** • **system-capabilities** • **system-description** • **system-name**

Step 3	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file, and remain within the configuration session.

Disabling LLDP Receive and Transmit Operations for an Interface

When you enable LLDP globally on the router, all supported interfaces are automatically enabled for LLDP receive and transmit operations. You can override this default by disabling these operations for a particular interface.

To disable LLDP receive and transmit operations for an interface, complete these steps:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **lldp**
4. **receive disable**
5. **transmit disable**

6. **end**
   or
   commit

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`interface GigabitEthernet 0/2/0/0`
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/2/0/0`
		Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. Possible interface types for this procedure are:
		• GigabitEthernet
		• TenGigE

Step 3	`lldp`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# lldp`
		(Optional) Enters LLDP configuration mode for the specified interface.

Step 4	`receive disable`
	Example:
	`RP/0/RSP0/CPU0:router(config-lldp)# receive disable`
		(Optional) Disables LLDP receive operations on the interface.

Step 5	**`transmit disable`**
	
	Example:
	`RP/0/RSP0/CPU0:router(config-lldp)#` **`transmit disable`**
		(Optional) Disables LLDP transmit operations on the interface.

Step 6	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config)#` **`end`**
	or
	`RP/0/RSP0/CPU0:router(config)#` **`commit`**
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file, and remain within the configuration session.

Verifying the LLDP Configuration

This section describes how to verify the LLDP configuration both globally, and for a particular interface.

Verifying the LLDP Global Configuration

To verify the LLDP global configuration status and operational characteristics, use the **show lldp** command as shown in this example:

```
RP/0/RSP0/CPU0:router# show lldp
Wed Apr 13 06:16:45.510 DST
Global LLDP information:
        Status: ACTIVE
        LLDP advertisements are sent every 30 seconds
        LLDP hold time advertised is 120 seconds
        LLDP interface reinitialisation delay is 2 seconds
```

If LLDP is not enabled globally, this output appears when you run the **show lldp** command:

```
RP/0/RSP0/CPU0:router# show lldp
Wed Apr 13 06:42:48.221 DST
% LLDP is not enabled
```

Verifying the LLDP Interface Configuration

To verify the LLDP interface status and configuration, use the **show lldp interface** command as shown in this example:

```
RP/0/RSP0/CPU0:router# show lldp interface GigabitEthernet 0/1/0/7
Wed Apr 13 13:22:30.501 DST


GigabitEthernet0/1/0/7:
        Tx: enabled
        Rx: enabled
        Tx state: IDLE
        Rx state: WAIT FOR FRAME
```

What To Do Next

To monitor and maintain LLDP on the system or get information about LLDP neighbors, use one of these commands:

Command	Description
clear lldp	Resets LLDP traffic counters or LLDP neighbor information
show lldp entry	Displays detailed information about LLDP neighbors
show lldp errors	Displays LLDP error and overflow statistics
show lldp neighbors	Displays information about LLDP neighbors
show lldp traffic	Displays statistics for LLDP traffic

Configuration Examples for Ethernet

This section provides the following configuration examples:

- Configuring an Ethernet Interface: Example, page 59
- Configuring MAC-Accounting: Example, page 60
- Configuring a Layer 2 VPN AC: Example, page 60
- Clock Interface Configuration: Example, page 60
- Enabling an Interface for Frequency Synchronization: Example, page 61
- Configuring LLDP: Examples, page 61

Configuring an Ethernet Interface: Example

This example shows how to configure an interface for a 10-Gigabit Ethernet modular services card:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/0/0/1
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RSP0/CPU0:router(config-if)# flow-control ingress
RP/0/RSP0/CPU0:router(config-if)# mtu 1448
RP/0/RSP0/CPU0:router(config-if)# mac-address 0001.2468.ABCD
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# end
```

```
Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# show interfaces TenGigE 0/0/0/1

TenGigE0/0/0/1 is down, line protocol is down
  Hardware is TenGigE, address is 0001.2468.abcd (bia 0001.81a1.6b23)
  Internet address is 172.18.189.38/27
  MTU 1448 bytes, BW 10000000 Kbit
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation ARPA,
  Full-duplex, 10000Mb/s, LR
  output flow control is on, input flow control is on
  loopback not set
  ARP type ARPA, ARP timeout 01:00:00
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

Configuring MAC-Accounting: Example

This example indicates how to configure MAC-accounting on an Ethernet interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RSP0/CPU0:router(config-if)# mac-accounting egress
RP/0/RSP0/CPU0:router(config-if)# commit
RP/0/RSP0/CPU0:router(config-if)# exit
RP/0/RSP0/CPU0:router(config)# exit
```

Configuring a Layer 2 VPN AC: Example

This example indicates how to configure a Layer 2 VPN AC on an Ethernet interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/RSP0/CPU0:router(config-if)# l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# l2protocol cpsv tunnel
RP/0/RSP0/CPU0:router(config-if-l2)# commit
```

Clock Interface Configuration: Example

```
RP/0/0/CPU0:ios#conf
RP/0/0/CPU0:ios(config)#clock-interface sync 0 location 0/0/CPU0
RP/0/0/CPU0:ios(config-clock-if)#frequency synchronization
RP/0/0/CPU0:ios(config-clk-freqsync)#selection input
RP/0/0/CPU0:ios(config-clk-freqsync)#commit
```

Enabling an Interface for Frequency Synchronization: Example

```
ios#conf
ios(config)#frequency synchronization
ios(config-freqsync)#commit
ios(config-freqsync)#exit
ios(config)#controller sonet 0/1/0/1
ios(config-sonet)#frequency synchronization
ios(config-sonet-freqsync)#wait-to-restore 0
ios(config-sonet-freqsync)#selection input
ios(config-sonet-freqsync)#commit
```

Configuring LLDP: Examples

This example shows how to enable LLDP globally on the router, and modify the default LLDP operational characteristics:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# lldp
RP/0/RSP0/CPU0:router(config)# lldp holdtime 60
RP/0/RSP0/CPU0:router(config)# lldp reinit 4
RP/0/RSP0/CPU0:router(config)# lldp timer 60
RP/0/RSP0/CPU0:router(config)# commit
```

This example shows how to disable a specific Gigabit Ethernet interface for LLDP transmission:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/2/0/0
RP/0/RSP0/CPU0:router(config-if)# lldp
RP/0/RSP0/CPU0:router(config-lldp)# transmit disable
```

Where to Go Next

When you have configured an Ethernet interface, you can configure individual VLAN subinterfaces on that Ethernet interface.

For information about modifying Ethernet management interfaces for the shelf controller (SC), route processor (RP), and distributed RP, see the Advanced Configuration and Modification of the Management Ethernet Interface on the Cisco ASR 9000 Series Router module later in this document.

For information about IPv6 see the *Implementing Access Lists and Prefix Lists on Cisco IOS XR Software* module in the *Cisco IOS XR IP Addresses and Services Configuration Guide.*

Additional References

The following sections provide references related to implementing Gigabit and 10-Gigabit Ethernet interfaces.

Related Documents

Related Topic	Document Title
Ethernet L2VPN	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*

Standards

Standards	Title
IEEE 802.1ag ITU-T Y.1731	*Ethernet OAM Connectivity Fault Management*

MIBs

MIBs	MIBs Link
IEEE CFM MIB	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Ethernet OAM on the Cisco ASR 9000 Series Router

This module describes the configuration of Ethernet Operations, Administration, and Maintenance (OAM) on the Cisco ASR 9000 Series Aggregation Services Routers.

Feature History for Configuring Ethernet OAM

Release	Modification
Release 3.7.2	Support for the following features was introduced: • Ethernet Link OAM • Ethernet CFM
Release 3.7.3	Support for the CFM Exploratory Linktrace feature was introduced.
Release 3.9.0	Support for the Ethernet SLA feature was introduced.
Release 3.9.1	Support for the following features was introduced: • Ethernet CFM on Link Aggregation Group (LAG) interfaces (Ethernet bundle interfaces), Ethernet and bundle subinterfaces, and LAG member (bundle member) interfaces. • EFD • AIS • Flexible tagging • The **ethernet cfm mep domain** command is replaced by the **ethernet cfm** and **mep domain** commands.

Release 4.0.0	Support for the following features was introduced: • The **action link-fault** command is replaced by the **action uni-directional link fault** command. • The **efd** keyword is added to put an interface into the line protocol down state, as an option for the following commands: – **action capabilities-conflict** – **action discovery-timeout** – **action session-down** – **action uni-directional link-fault** • Uni-directional link-fault detection to identify local link-faults and send notification to a remote Ethernet OAM peer using the **uni-directional link-fault detection** command. • Support for the following enhancements to Ethernet SLA was added: – Support for on-demand Ethernet SLA operations using the **ethernet sla on-demand operation** commands. – One-way delay and jitter measurements using the following new keyword options for the **statistics measure** command: **one-way-delay-ds**. **one-way-delay-sd**. **one-way-jitter-ds**. **one-way-jitter-sd** – Specification of a test pattern to pad loopback packets when measuring delay. – Displaying the time when the minimum (Min) and maximum (Max) values of a statistic occurred in the measurement time period in the **show ethernet sla statistics detail** command.
Release 4.0.1	Support for Ethernet CFM on Multi-Chassis Link Aggregation Groups (MC-LAG) was added.
Release 4.1.0	Support for the following feature was introduced: • E-LMI • Timestamps for delay packets were changed from being derived by the sytem time-of-day (NTP) clock to the DTI timing input on the clock-interfaces on the RSP. • CFM Y.1731 ITU Carrier Code (ICC)-based MEG ID (MAID) format.
Release 4.2.0	Support for Unidirectional Link Detection Protocol (UDLD) was added.
Release 4.2.2	Support for Unidirectional Link Routing (UDLR) was included.

Contents

Prerequisites for Configuring Ethernet OAM

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring Ethernet OAM, confirm that at least one of the Gigabit Ethernet line cards supported on the router is installed:

- 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination line card (A9K-2T20GE-B and A9K-2T20GE-L)
- 4-Port 10-Gigabit Ethernet line card (A9K-4T-L, -B, or -E)
- 8-Port 10-Gigabit Ethernet DX line card (A9K-8T/4-L, -B, or -E)
- 8-Port 10-Gigabit Ethernet line card (A9K-8T-L, -B, or -E)
- 16-Port 10-Gigabit Ethernet SFP+ line card (A9K-16T/8-B and A9K-16T/8-B+AIP)
- 40-Port Gigabit Ethernet line card (A9K-40GE-L, -B, or -E)

Information About Configuring Ethernet OAM

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

- Ethernet Link OAM, page 68
- Ethernet CFM, page 70
- Ethernet SLA (Y.1731 Performance Monitoring), page 89
- Ethernet LMI, page 93
- Unidirectional Link Detection Protocol, page 97
- Unidirectional Link Routing, page 99

Ethernet Link OAM

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

Ethernet link OAM can be configured in the following ways:

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

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

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

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

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

The following standard Ethernet Link OAM features are supported on the router:

- Neighbor Discovery, page 69
- Link Monitoring, page 69
- MIB Retrieval, page 69
- Miswiring Detection (Cisco-Proprietary), page 69
- Remote Loopback, page 69
- SNMP Traps, page 69
- Unidirectional Link Fault Detection, page 69

Neighbor Discovery

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

Link Monitoring

Link monitoring enables an OAM peer to monitor faults that cause the quality of a link to deteriorate over time. When link monitoring is enabled, an OAM peer can be configured to take action when the configured thresholds are exceeded.

MIB Retrieval

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

Miswiring Detection (Cisco-Proprietary)

Miswiring Detection is a Cisco-proprietary feature that uses the 32-bit vendor field in every Information OAMPDU to identify potential miswiring cases.

Remote Loopback

Remote loopback enables one side of a link to put the remote side of the link into loopback mode for testing. When remote loopback is enabled, all packets initiated by the master side of the link are looped back to the master side, unaltered by the remote (slave) side. In remote loopback mode, the slave side is not allowed to inject any data into the packets.

SNMP Traps

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

Unidirectional Link Fault Detection

Unidirectional link fault detection describes an Ethernet link OAM function that runs directly on physical Ethernet interfaces (not VLAN subinterfaces or bundles) that uses a defined link fault message to signal link faults to a remote host. Unidirectional link fault detection offers similar functionality to Gigabit Ethernet and Ten Gigabit Ethernet hardware-level signaling of a link fault, but it is done at a higher protocol layer as part of Ethernet link OAM. The hardware function uses the Remote Fault Indication bit set in a frame that is signaled out-of-band, where unidirectional link fault detection signals the error using an OAMPDU.

Unidirectional link fault detection only applies to a single, physical link. When the remote host receives the link fault message, the interface can be shut down for all higher-layer protocols, and specifically, Layer 2 switching and Layer 3 routing protocols. While the fault is detected, a link fault message is sent periodically to the remote host. Once a fault is no longer detected, the link fault message is no longer sent, and the remote host can bring back the interface.

Unidirectional link fault detection is configured using the **uni-directional link-fault detection** command, and does not affect how the receipt of link-fault messages are handled by the router. Actions to be taken for the receipt of link-fault messages are configured using the **action uni-directional link-fault** command.

Ethernet CFM

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

CFM is defined in two standards:

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

Ethernet CFM on the Cisco ASR 9000 Series Router supports the following functions of ITU-T Y.1731:

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

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

- ETH-AIS—The reception of ETH-LCK messages is also supported.
- ETH-DM—This is supported with the Ethernet SLA feature. For more information about Ethernet SLA, see the "Ethernet SLA (Y.1731 Performance Monitoring)" section on page 89.

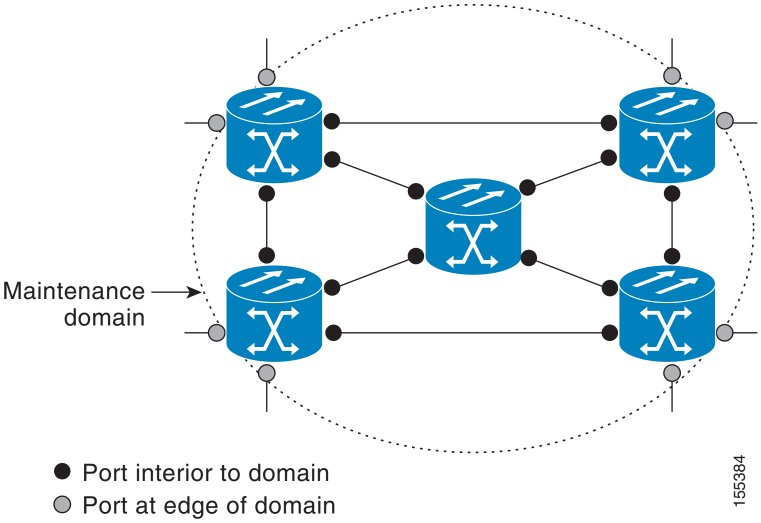
To understand how the CFM maintenance model works, you need to understand the following concepts and features:

- Maintenance Domains, page 71
- Services, page 73
- Maintenance Points, page 73
- CFM Protocol Messages, page 76
- MEP Cross-Check, page 83
- Configurable Logging, page 84
- EFD, page 84
- Flexible VLAN Tagging for CFM, page 85
- CFM on MC-LAG, page 86

Maintenance Domains

A *maintenance domain* describes a management space for the purpose of managing and administering a network. A domain is owned and operated by a single entity and defined by the set of interfaces internal to it and at its boundary, as shown in Figure 1.

Figure 1 CFM Maintenance Domain



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

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

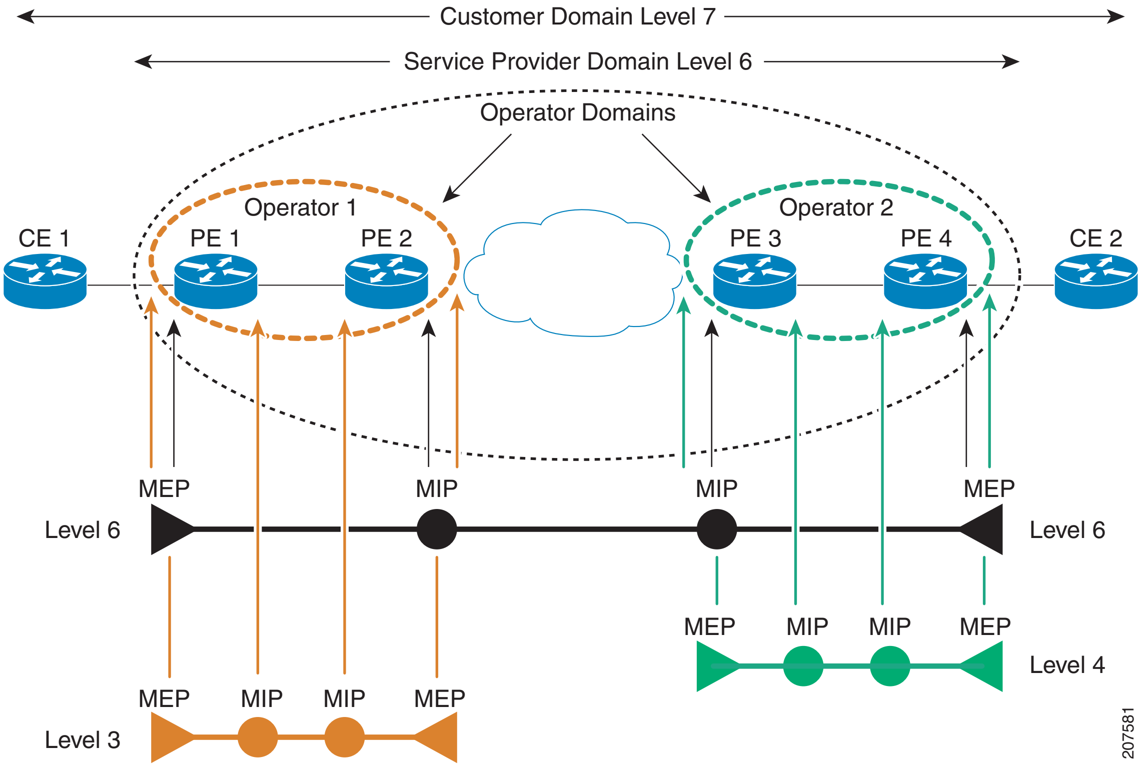
- The customer can use CFM between their CE devices, to verify and manage connectivity across the whole network.
- The service provider can use CFM between their PE devices, to verify and manage the services they are providing.
- Each operator can use CFM within their operator network, to verify and manage connectivity within their network.

Each organization uses a different CFM maintenance domain.

Figure 2 shows an example of the different levels of maintenance domains in a network.

Note In CFM diagrams, the conventions are that triangles represent MEPs, pointing in the direction that the MEP sends CFM frames, and circles represent MIPs. For more information about MEPs and MIPs, see the "Maintenance Points" section on page 73.

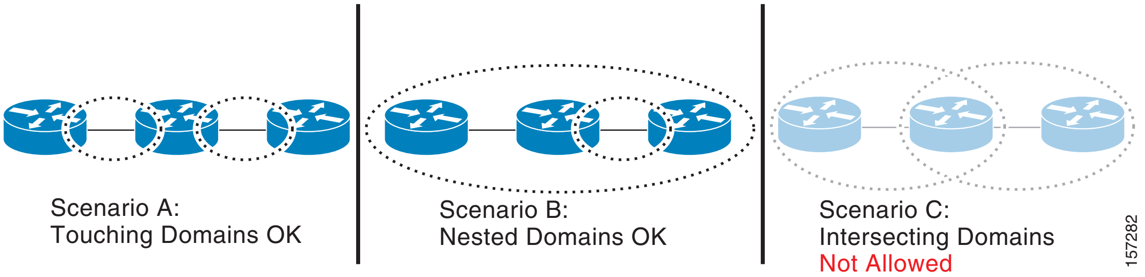
Figure 2 Different CFM Maintenance Domains Across a Network



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

CFM maintenance domains may touch or nest, but cannot intersect. Figure 3 illustrates the supported structure for touching and nested domains, and the unsupported intersection of domains.

Figure 3 Supported CFM Maintenance Domain Structure



Services

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

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

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

Maintenance Points

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

A maintenance point is always associated with a particular CFM service, and therefore with a particular maintenance domain at a particular level. Maintenance points generally only process CFM frames at the same level as their associated maintenance domain. Frames at a higher maintenance level are always forwarded transparently, while frames at a lower maintenance level are normally dropped. This helps enforce the maintenance domain hierarchy described in the "Maintenance Domains" section on page 71, and ensures that CFM frames for a particular domain cannot leak out beyond the boundary of the domain.

There are two types of MP:

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

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

MIP Creation

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

- The bridge-domain or cross-connect for the interface is found, and all services associated with that bridge-domain or cross-connect are considered for MIP auto-creation.
- The level of the highest-level MEP on the interface is found. From among the services considered above, the service in the domain with the lowest level that is higher than the highest MEP level is selected. If there are no MEPs on the interface, the service in the domain with the lowest level is selected.
- The MIP auto-creation configuration (**mip auto-create** command) for the selected service is examined to determine whether a MIP should be created.

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

MEP and CFM Processing Overview

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

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

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

Figure 4 illustrates the monitored areas for Down and Up MEPs.

Figure 4 Monitored Areas for Down and Up MEPs

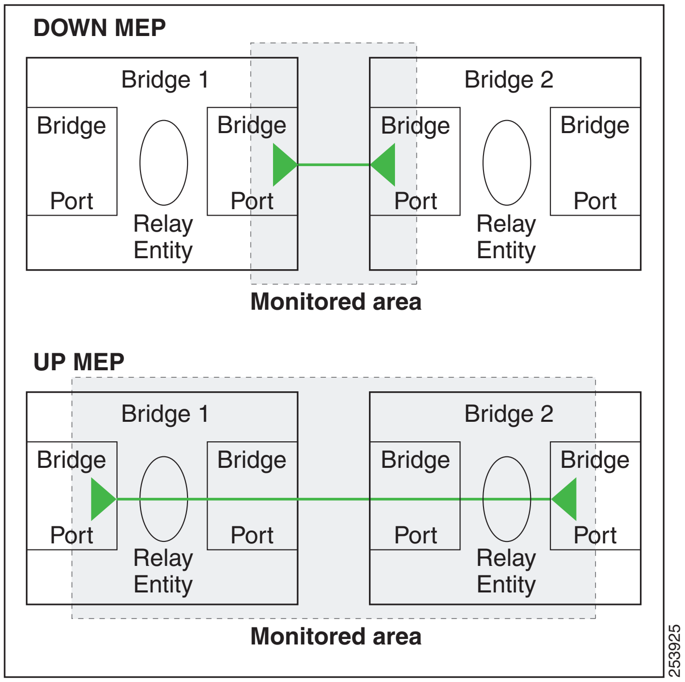
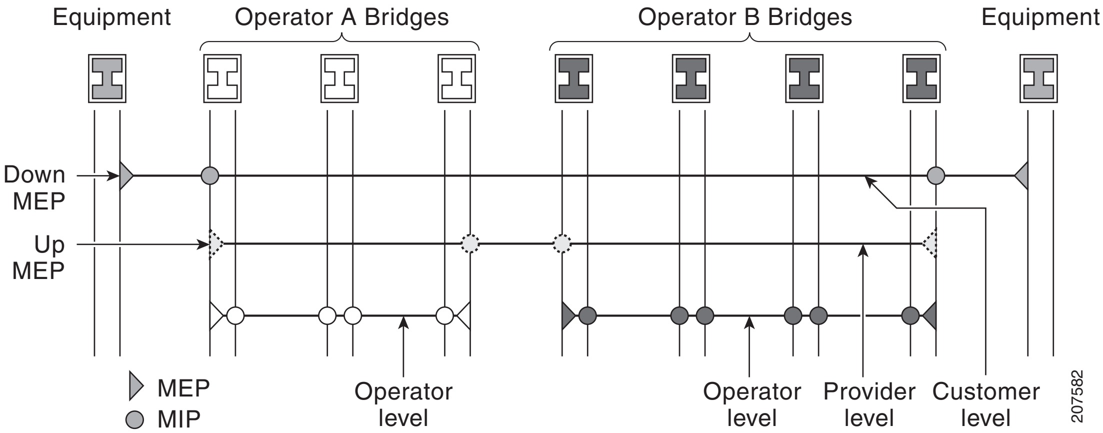


Figure 5 shows maintenance points at different levels. Because domains are allowed to nest but not intersect (see Figure 3), a MEP at a low level always corresponds with a MEP or MIP at a higher level. In addition, only a single MIP is allowed on any interface—this is generally created in the lowest domain that exists at the interface and that does not have a MEP.

Figure 5 CFM Maintenance Points at Different Levels



MIPs and Up MEPs can only exist on switched (Layer 2) interfaces, because they send and receive frames from the bridge relay function. Down MEPs can be created on switched (Layer 2) or routed (Layer 3) interfaces.

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

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

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

CFM Protocol Messages

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

This section describes the following CFM messages:

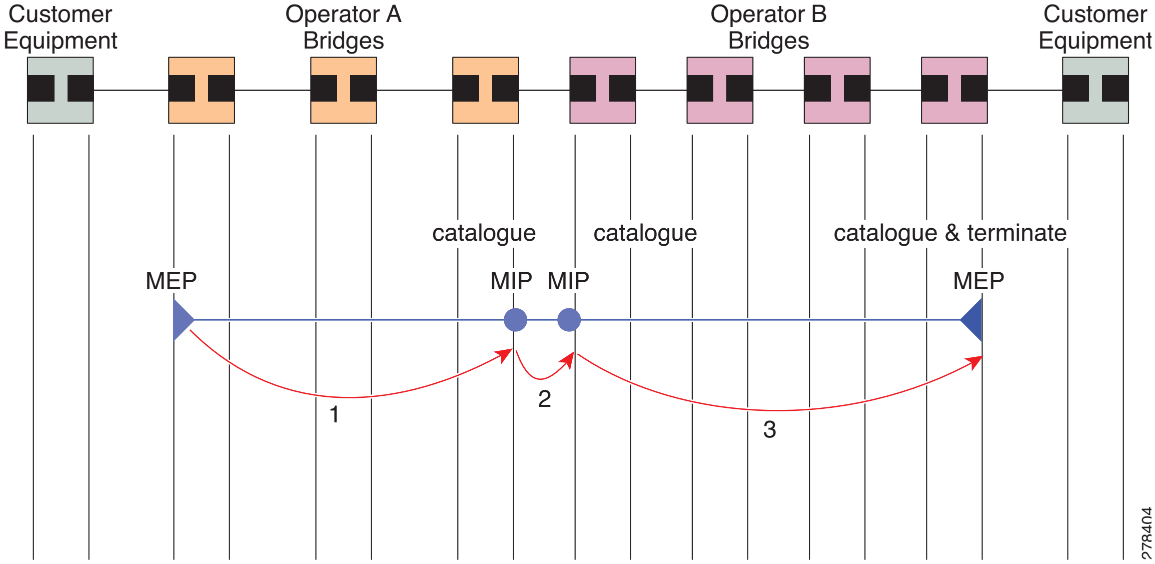
- Continuity Check (IEEE 802.1ag and ITU-T Y.1731), page 76
- Loopback (IEEE 802.1ag and ITU-T Y.1731), page 78
- Linktrace (IEEE 802.1ag and ITU-T Y.1731), page 79
- Exploratory Linktrace (Cisco), page 81
- Alarm Indication Signal (ITU-T Y.1731), page 82
- Delay and Jitter Measurement (ITU-T Y.1731), page 83

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

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

MIPs also receive CCMs. MIPs use the information to build a MAC learning database that is used when responding to Linktrace. For more information about Linktrace, see the "Linktrace (IEEE 802.1ag and ITU-T Y.1731)" section on page 79.

Figure 6 Continuity Check Message Flow



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

- 100ms
- 1s
- 10s
- 1 minute
- 10 minutes

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

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

- A configured identifier for the domain of the transmitting MEP. This is referred to as the Maintenance Domain Identifier (MDID).
- A configured identifier for the service of the transmitting MEP. This is referred to as the Short MA Name (SMAN). Together, the MDID and the SMAN make up the Maintenance Association Identifier (MAID). The MAID must be configured identically on every MEP in the service.
- A configured numeric identifier for the MEP (the MEP ID). Each MEP in the service must be configured with a different MEP ID.
- A sequence number.
- A Remote Defect Indication (RDI). Each MEP includes this in the CCMs it is sending, if it has detected a defect relating to the CCMs it is receiving. This notifies all the MEPs in the service that a defect has been detected somewhere in the service.
- The interval at which CCMs are being transmitted.
- The status of the interface where the MEP is operating—for example, whether the interface is up, down, STP blocked, and so on.

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

The following defects can be detected from received CCMs:

- Interval mismatch—The CCM interval in the received CCM does not match the interval that the MEP is sending CCMs.
- Level mismatch—A MEP has received a CCM carrying a lower maintenance level than the MEPs own level.
- Loop—A CCM is received with the source MAC address equal to the MAC address of the interface where the MEP is operating.
- Configuration error—A CCM is received with the same MEP ID as the MEP ID configured for the receiving MEP.
- Cross-connect—A CCM is received with an MAID that does not match the locally configured MAID. This generally indicates a VLAN misconfiguration within the network, such that CCMs from one service are leaking into a different service.
- Peer interface down—A CCM is received that indicates the interface on the peer is down.
- Remote defect indication—A CCM is received carrying a remote defect indication.

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

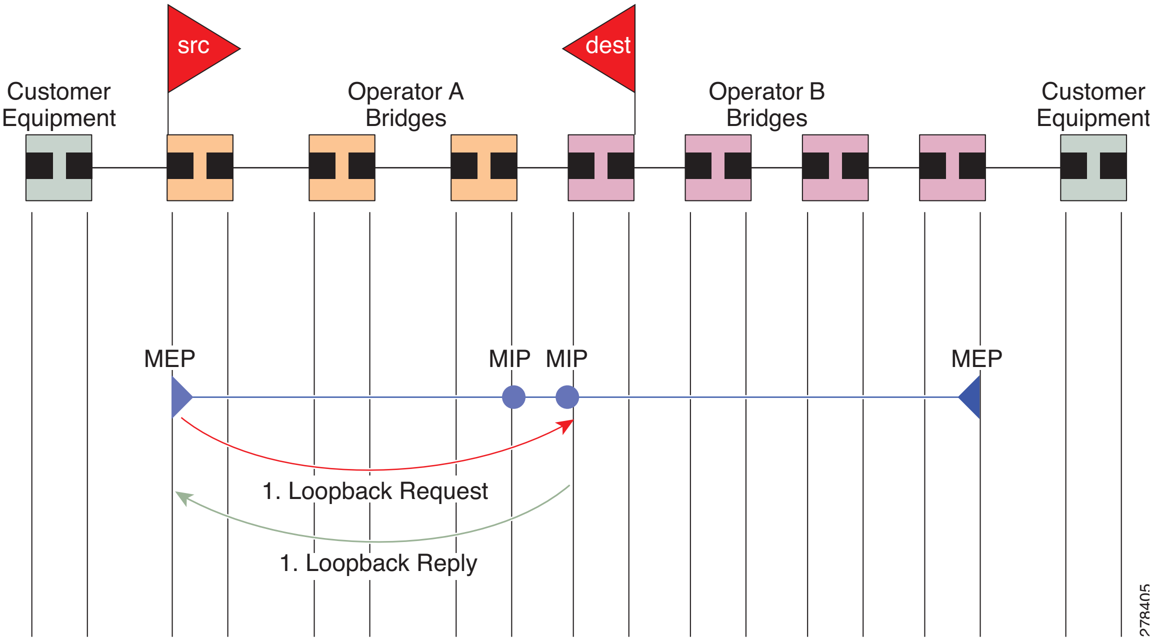
Out-of-sequence CCMs can also be detected by monitoring the sequence number in the received CCMs from each peer MEP. However, this is not considered a CCM defect.

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

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

Figure 7 shows an example of CFM loopback message flow between a MEP and MIP.

Figure 7 Loopback Messages



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

Except for one-way delay and jitter measurements, loopback messages can also be used for Ethernet SLA, if the peer does not support delay measurement.

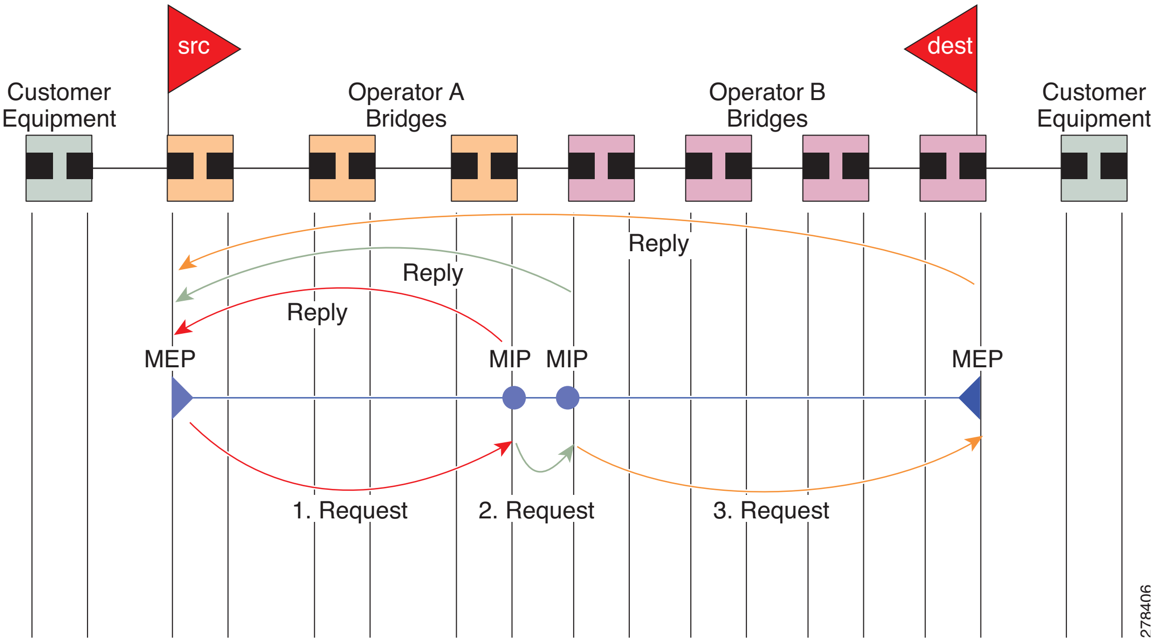
Note The Ethernet CFM loopback function should not be confused with the remote loopback functionality in Ethernet Link OAM (see the "Remote Loopback" section on page 69). CFM loopback is used to test connectivity with a remote MP, and only the CFM LBM packets are reflected back, but Ethernet Link OAM remote loopback is used to test a link by taking it out of normal service and putting it into a mode where it reflects back all packets.

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

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

Figure 8 shows an example of CFM linktrace message flow between MEPs and MIPs.

Figure 8 Linktrace Message Flow



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

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

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

1. The target MAC address in the LTM is looked up in the bridge MAC learning table. If the MAC address is known, and therefore the egress interface is known, then an LTR is sent.
2. If the MAC address is not found in the bridge MAC learning table, then it is looked up in the CCM learning database. If it is found, then an LTR is sent.
3. If the MAC address is not found, then no LTR is sent (and the LTM is not forwarded).

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

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

Exploratory Linktrace (Cisco)

Exploratory Linktrace is a Cisco extension to the standard linktrace mechanism described above. It has two primary purposes:

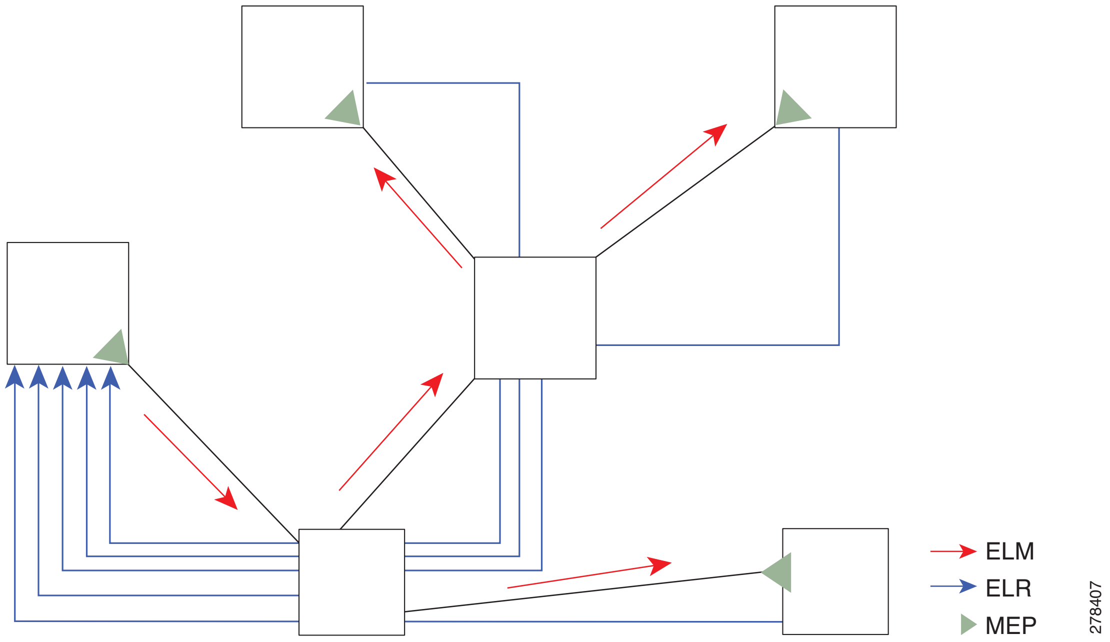
- Provide a mechanism to locate faults in cases where standard linktrace does not work, such as when a MAC address has never been seen previously in the network. For example, if a new MEP has been provisioned but is not working, standard linktrace does not help isolate a problem because no frames will ever have been received from the new MEP. Exploratory Linktrace overcomes this problem.
- Provide a mechanism to map the complete active network topology from a single node. This can only be done currently by examining the topology (for example, the STP blocking state) on each node in the network individually, and manually combining this information to create the overall active topology map. Exploratory linktrace allows this to be done automatically from a single node.

Exploratory Linktrace is implemented using the Vendor Specific Message (VSM) and Vendor Specific Reply (VSR) frames defined in ITU-T Y.1731. These allow vendor-specific extensions to be implemented without degrading interoperability. Exploratory Linktrace can safely be deployed in a network that includes other CFM implementations because those implementations will simply ignore the Exploratory Linktrace messages.

Exploratory Linktrace is initiated at the request of the administrator, and results in the local MEP sending a multicast Exploratory Linktrace message. Each MP in the network that receives the message sends an Exploratory Linktrace reply. MIPs that receive the message also forward it on. The initiating MEP uses all the replies to create a tree of the overall network topology.

Figure 9 show an example of the Exploratory Linktrace message flow between MEPs.

Figure 9 Exploratory Linktrace Messages and Replies



To avoid overloading the originating MEP with replies in a large network, responding MPs delay sending their replies for a random amount of time, and that time increases as the size of the network increases.

In a large network, there will be a corresponding large number of replies and the resulting topology map will be equally large. If only a part of the network is of interest, for example, because a problem has already been narrowed down to a small area, then the Exploratory Linktrace can be “directed” to start at a particular MP. Replies will thus only be received from MPs beyond that point in the network. The replies are still sent back to the originating MEP.

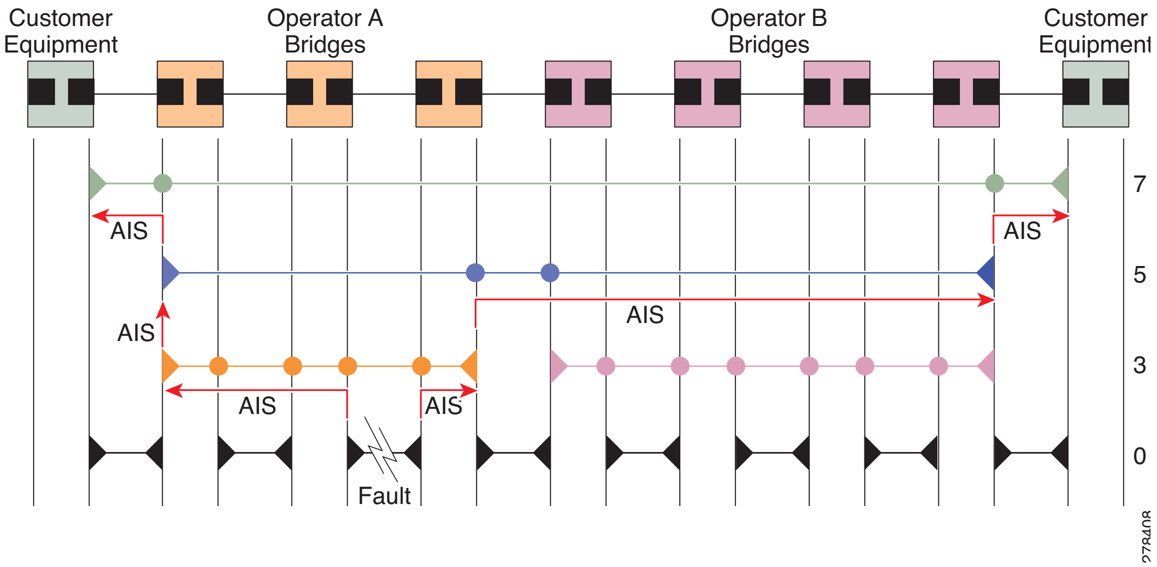
Alarm Indication Signal (ITU-T Y.1731)

Alarm Indication Signal (AIS) messages are used to rapidly notify MEPs when a fault is detected in the middle of a domain, in an event driven way. MEPs thereby learn of the fault much sooner than if they relied on detecting a loss of continuity, for example, failure to receive some number of consecutive CCMs.

Unlike all other CFM messages, AIS messages are injected into the middle of a domain, and sent outward toward the MEPs at the edge of the domain. Typically, AIS messages are injected by a MEP in a lower level domain. To put it another way, when a MEP sends AIS messages, they are sent in the opposite direction to other CFM messages sent by the MEP, and at a level above the MEP’s own level. The AIS messages are received by the MEPs in the higher level domain, not by the peer MEPs in the same domain as the MEP sending the AIS. When a MEP receives an AIS message, it may itself send another AIS message at an even higher level.

Figure 10 show an example of AIS message flow. The maintenance domain levels are numbered at the right side of the diagram.

Figure 10 AIS Message Flow



AIS is only applicable in point-to-point networks. In multipoint networks with redundant paths, a failure at a low level does not necessarily result in a failure at a higher level, as the network may reconverge so as to route around the failed link.

AIS messages are typically sent by a MEP. However, AIS messages can also be sent when there is no MEP present, if a fault is detected in the underlying transport, such as if an interface goes down. In ITU-T Y.1731 these are referred to as *server MEPs*.

AIS messages are sent in response to a number of failure conditions:

- Detection of CCM defects, as described “Continuity Check (IEEE 802.1ag and ITU-T Y.1731)” section on page 76.

- Loss of continuity.
- Receipt of AIS messages.
- Failure in the underlying transport, such as when an interface is down.

Received AIS messages can be used to detect and act on failures more quickly than waiting for a loss of continuity. They can also be used to suppress any failure action, on the basis that the failure has already been detected at a lower level and will be handled there. This is described in ITU-T Y.1731; however, the former is often more useful.

Delay and Jitter Measurement (ITU-T Y.1731)

The router supports one-way and two-way delay measurement using two packet types:

- Delay Measurement Message (DMM)
- Delay Measurement Response (DMR)

These packets are unicast similar to loopback messages. The packets carry timestamps generated by the system time-of-day clock to support more accurate delay measurement, and also support an SLA manageability front-end. Beginning in Cisco IOS XR Release 4.1, the DDM & DDR packets carry timestamps derived from the DTI timing input on the clock-interface port on the RSP.

However, unlike loopback messages, these message types can also measure one-way delay and jitter either from destination to source, or from source to destination.

For more information about SLA, see the "Ethernet SLA (Y.1731 Performance Monitoring)" section on page 89.

MEP Cross-Check

MEP cross-check supports configuration of a set of expected peer MEPs so that errors can be detected when any of the known MEPs are missing, or if any additional peer MEPs are detected that are not in the expected group.

The set of expected MEP IDs in the service is user-defined. Optionally, the corresponding MAC addresses can also be specified. CFM monitors the set of peer MEPs from which CCMs are being received. If no CCMs are ever received from one of the specified expected peer MEPs, or if a loss of continuity is detected, then a cross-check "missing" defect is detected. Similarly, if CCMs are received from a matching MEP ID but with the wrong source MAC address, a cross-check "missing" defect is detected. If CCMs are subsequently received that match the expected MEP ID, and if specified, the expected MAC address, then the defect is cleared.

Note While loss of continuity can be detected for any peer MEP, it is only treated as a defect condition if cross-check is configured.

If cross-check is configured and CCMs are received from a peer MEP with a MEP ID that is not expected, this is detected as a cross-check "unexpected" condition. However, this is not treated as a defect condition.

Configurable Logging

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

- New peer MEPs are detected, or loss of continuity with a peer MEP occurs.
- Changes to the CCM defect conditions are detected.
- Cross-check “missing” or “unexpected” conditions are detected.
- AIS condition detected (AIS messages received) or cleared (AIS messages no longer received).
- EFD used to shut down an interface, or bring it back up.

EFD

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

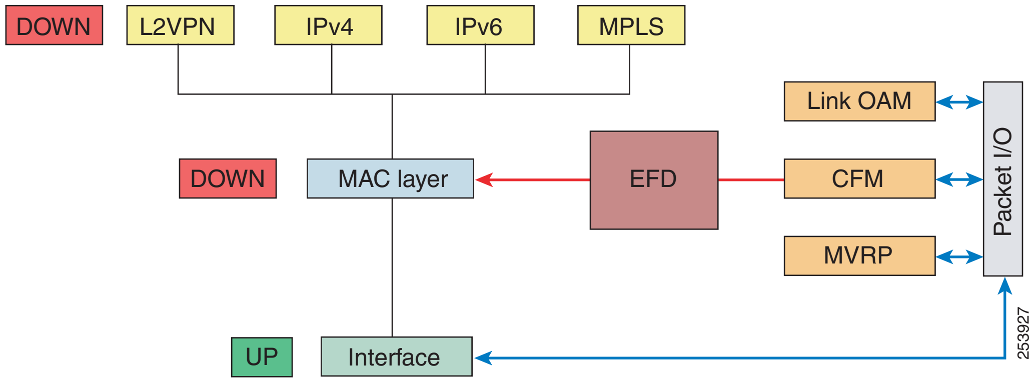
Unlike many other interface types, Ethernet interfaces do not have a line protocol, whose state is independent from that of the interface. For Ethernet interfaces, this role is handled by the physical-layer Ethernet protocol itself, and therefore if the interface is physically up, then it is available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

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

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

Figure 11 shows CFM detection of an error on one of its sessions EFD signaling an error to the corresponding MAC layer for the interface. This triggers the MAC to go to a down state, which further triggers all higher level protocols (Layer 2 pseudowires, IP protocols, and so on) to go down and also trigger a reconvergence where possible. As soon as CFM detects there is no longer any error, it can signal to EFD and all protocols will once again go active.

Figure 11 CFM Error Detection and EFD Trigger



Flexible VLAN Tagging for CFM

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

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

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

Figure 12 shows an example of a network with multiple VLANS using CFM.

Figure 12 Service Provider Network With Multiple VLANs and CFM

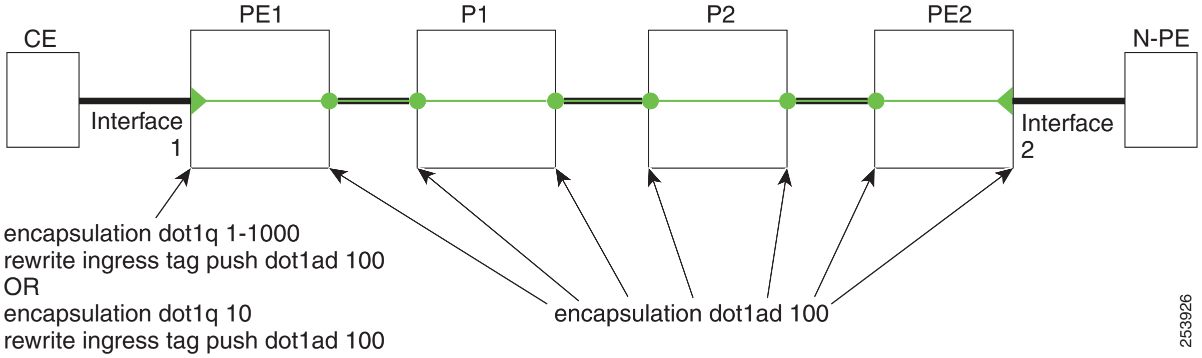

PE1
P1
P2
PE2
CE
N-PE
Interface 1
Interface 2
encapsulation dot1q 1-1000
rewrite ingress tag push dot1ad 100
OR
encapsulation dot1q 10
rewrite ingress tag push dot1ad 100
encapsulation dot1ad 100
253926

Figure 12 shows a provider's access network, where the S-VLAN tag is used as the service delimiter. PE1 faces the customer, and PE2 is at the edge of the access network facing the core. N:1 bundling is used, so the interface encapsulation matches a range of C-VLAN tags. This could potentially be the full range, resulting in all:1 bundling. There is also a use case where only a single C-VLAN is matched, but the S-VLAN is nevertheless used as the service delimiter—this is more in keeping with the IEEE model, but limits the provider to 4094 services.

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

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

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

CFM on MC-LAG

CFM on Multi-Chassis Link Aggregation Groups is supported on the Cisco ASR 9000 Series Router in the following typical network environment:

- The customer edge (CE) device is a dual-homed device that is connected to two provider edge (PE) point-of-attachment (POA) devices. However, the dual-homed device operates without awareness of connectivity to multiple PEs.
- The two points of attachment at the PE form a redundancy group (RG), with one POA functioning as the active POA, and the other as the standby POA for the dual-homed device link.
- As with typical failover scenarios, if a failure occurs with the active POA, the standby POA takes over to retain the dual-homed device's connectivity to the network.

CFM on MC-LAG support can be qualified at two levels:

- CFM for the RG level—CFM context is per redundancy group and verifies connectivity for the entire RG.
- CFM for the POA level—CFM context is per point of attachment and verifies connectivity to a single POA.

Both levels of CFM support have certain restrictions and configuration guidelines that you must consider for successful implementation.

This section includes the following topics:

- RG-Level CFM, page 87
- POA-Level CFM, page 88
- Supported Features for CFM on MC-LAG, page 88
- Restrictions for CFM on MC-LAG, page 88

For more information about LAG and MC-LAG on the Cisco ASR 9000 Series Router, see the "Configuring Link Bundling on the Cisco ASR 9000 Series Router" chapter in this guide.

RG-Level CFM

RG-level CFM is comprised of three areas of monitoring:

- RG Downlink Monitoring, page 87
- RG Uplink Monitoring, page 87
- End-to-End Service Monitoring, page 87

RG Downlink Monitoring

RG downlink monitoring uses CFM to verify connectivity between the dual-homed device and the RG.

To configure RG downlink monitoring, be sure that the following requirements are met:

- Down MEPs are configured on the bundle.
- Down MEPs on each POA are configured identically, using the same MEP ID and source MAC address.

This configuration has the following restrictions:

- The CCM loss time is greater than the failover time (typically 50 ms), due to the shortest CCM interval of 100 ms that is currently supported, which results in the shortest CCM loss time of 350 ms.

RG Uplink Monitoring

RG uplink monitoring uses CFM to verify connectivity from the active POA to the core.

To configure RG uplink monitoring, be sure that the following requirements are met:

- Up MEPs are configured on the bundle interface or bundle subinterface on each POA.
- Up MEPs on each POA are configured identically, using the same MEP ID and source MAC address.

End-to-End Service Monitoring

End-to-end service monitoring uses CFM to verify the end-to-end service between the dual-homed devices.

To configure end-to-end service monitoring, be sure that the following requirements are met:

- A down MEP is configured on the dual-homed device bundle interface or bundle subinterface.

- If optional MIPs are configured, then each POA is configured with a MIP on the bundle.
- Each POA can have a MIP on the uplink interface (if native Ethernet is used).
- The active and standby POA is configured identically.

This configuration has the following restrictions:

- The MIP on the standby POA will not respond to loopback or linktrace requests.

POA-Level CFM

POA-level monitoing uses CFM to verify connectivity between the dual-homed device and a single POA.

To configure POA-level CFM, be sure that the following requirements are met:

- Down MEPs are configured on bundle members only.

This configuration has the following restrictions:

- POA-level monitoring is not supported on uplinks between a single POA and the core.

Supported Features for CFM on MC-LAG

CFM on MC-LAG supports the following CFM features:

- All existing IEEE 802.1ag and Y.1731 functionality on the Cisco ASR 9000 Series Router is supported on an MC-LAG RG.
- CFM maintenance points are supported on an MC-LAG interface. Maintenance points on a standby link are put into standby state.
- Maintenance points in standby state receive CFM messages, but do not send or reply to any CFM messages.
- When a MEP transitions from active to standby, all CCM defects and alarms are cleared.
- Standby MEPs record remote MEP errors and timeouts, but do not report faults. This means that remote MEPs and their errors will appear in **show** commands, but no logs, alarms, MIB traps, or EFD are triggered and AIS messages are not sent.
- When a MEP transitions from standby to active, any CCM defects previously detected while the MEP was in standby are reapplied and immediate actions are taken (logs, alarms, MIB traps, EFD, and so on).
- CFM on MC-LAG supports the same scale for bundle interfaces that is supported on the Cisco ASR 9000 Series Router.

Restrictions for CFM on MC-LAG

To support CFM on MC-LAG, you must consider the following restrictions and requirements:

- The CFM configuration must be the same on both the active and standby POAs.
- The CFM state is not synchronized between the two POAs. This can lead to flapping of the interface line protocol state on POA failover if EFD is configured. Fault alarms might also be delayed if a failover occurs just after a fault has been detected.
- POA-level CFM monitoring is not supported on a native Ethernet uplink interface.
- MEPs on bundle interfaces at level 0 are not supported.
- Loopback, linktrace, and Y.1731 SLA operations cannot be started from a MEP in standby state.

- Checks for configuration consistency of MEP IDs to ensure identical configuration of POAs is not supported.
- Y.1731 SLA statistics can be split between the two POAs if a failover occurs. An external network management system would need to collect and collate these statistics from the two POAs.

Ethernet SLA (Y.1731 Performance Monitoring)

Customers require their service providers to conform to a Service Level Agreement (SLA). Consequently, service providers must be able to monitor the performance characteristics of their networks. Likewise, customers also want to monitor the performance characteristics of their networks. Cisco provides Y.1731 performance monitoring using the Cisco Ethernet SLA feature.

An SLA defines a set of criteria that guarantees a minimum level of service for customers using a service provider network. The criteria can cover many different areas, including latency, jitter, frame loss, and availability.

The Cisco Ethernet SLA feature conforms to the following standards:

- IEEE 802.1ag
- ITU-T Y.1731

The Cisco Ethernet SLA feature provides the architecture to monitor a network at Layer 2. This architecture provides functions such as collecting, storing, displaying, and analyzing SLA statistics. These SLA statistics can be stored and displayed in various ways, so that statistical analysis can be performed.

Ethernet SLA provides the framework for performing the following major functions of performance monitoring:

- Sending probes consisting of one or more packets to measure performance

  Ethernet SLA provides a flexible mechanism for sending SLA probes to measure performance. Probes can consist of either CFM loopback or CFM delay measurement packets. Options are available to modify how often the packets are sent, and to specify the attributes of the probe packets such as the size and priority.

- Scheduling of operations consisting of periodic probes.

  A flexible mechanism is provided by Ethernet SLA to specify how often each probe should be executed, how long it should last, and when the first probe should start. Probes can be scheduled to run back-to-back to provide continuous measurements, or at a defined interval ranging from once a minute to once a week.

- Collecting and storing results.

  Ethernet SLA provides flexibility to specify which performance parameters should be collected and stored for each measurement probe. Performance parameters include frame delay and jitter (inter-frame delay variation). For each performance parameter, either each individual result can be stored, or the results can be aggregated by storing a counter of the number of results that fall within a particular range. A configurable amount of historical data can also be stored as well as the latest results.

- Analyzing and displaying results.

Ethernet SLA performs some basic statistical analysis on the collected results, such as calculating the minimum, maximum, mean and standard deviation. It also records whether any of the probe packets were lost or misordered, or if there is any reason why the results may not be a true reflection of the performance (for example if a big jump in the local time-of-day clock was detected during the time when the measurements were being made).

Ethernet SLA Concepts

To successfully configure the Cisco Ethernet SLA feature, you should understand the following concepts:

- Ethernet SLA Statistic, page 90
- Ethernet SLA Measurement Packet, page 90
- Ethernet SLA Sample, page 91
- Ethernet SLA Probe, page 91
- Ethernet SLA Burst, page 91
- Ethernet SLA Schedule, page 91
- Ethernet SLA Bucket, page 91
- Ethernet SLA Aggregation Bin, page 92
- Ethernet SLA Operation Profile, page 92
- Ethernet SLA Operation, page 92
- Ethernet SLA On-Demand Operation, page 92

Ethernet SLA Statistic

A *statistic* in Ethernet SLA is a single performance parameter. The following statistics can be measured by Ethernet SLA:

- Round-trip delay
- Round-trip jitter
- One-way delay from source to destination
- One-way jitter from source to destination
- One-way delay from destination to source
- One-way jitter from destination to source

Note Not all statistics can be measured by all types of packet. For example, one-way statistics cannot be measured when using CFM loopback packets.

Ethernet SLA Measurement Packet

An Ethernet SLA *measurement packet* is a single protocol message and corresponding reply that is sent on the network for the purpose of making SLA measurements. The following types of measurement packet are supported:

- CFM Delay Measurement (Y.1731 DMM/DMR packets)—CFM delay measurement packets contain timestamps within the packet data that can be used for accurate measurement of frame delay and jitter. These packets can be used to measure round-trip or one-way statistics; however, the size of the DMM/DMR packets cannot be modified.
- CFM loopback (LBM/LBR)—CFM loopback packets are less accurate, but can be used if the peer device does not support DMM/DMR packets. Only round-trip statistics can be measured because these packets do not contain timestamps. However, loopback packets can be padded, so measurements can be made using frames of a specific size.

Ethernet SLA Sample

A *sample* is a single result—a number—that relates to a given statistic. For some statistics such as round-trip delay, a sample can be measured using a single measurement packet. For other statistics such as jitter, obtaining a sample requires two measurement packets.

Ethernet SLA Probe

A *probe* is a sequence of measurement packets used to gather SLA samples for a specific set of statistics. The measurement packets in a probe are of a specific type (for example, CFM delay measurement or CFM loopback) and have specific attributes, such as the frame size and priority.

Note A single probe can collect data for different statistics at the same time, using the same measurement packets (for example, one-way delay and round-trip jitter).

Ethernet SLA Burst

Within a probe, measurement packets can either be sent individually, or in bursts. A *burst* contains two or more packets sent within a short interval apart. Each burst can last up to one minute, and bursts can follow each other immediately to provide continuous measurement within the probe.

For statistics that require two measurement packets for each sample (such as jitter), samples are only calculated based on measurement packets in the same burst. For all statistics, it is more efficient to use bursts than to send individual packets.

Ethernet SLA Schedule

An Ethernet SLA *schedule* describes how often probes are sent, how long each probe lasts, and at what time the first probe starts.

Ethernet SLA Bucket

For a particular statistic, a *bucket* is a collection of results that were gathered during a particular period of time. All of the samples for measurements that were initiated during the period of time represented by a bucket are stored in that bucket. Buckets allow results from different periods of time to be compared (for example, peak traffic to off-peak traffic).

By default, a separate bucket is created for each probe; that is, the bucket represents the period of time starting at the same time as the probe started, and continuing for the duration of the probe. The bucket will therefore contain all the results relating to measurements made by that probe.

Ethernet SLA Aggregation Bin

Rather than storing each sample separately within a bucket, an alternative is to aggregate the samples into bins. An *aggregation bin* is a range of sample values, and contains a counter of the number of samples that were received that fall within that range. The set of bins forms a histogram. When aggregation is enabled, each bucket contains a separate set of bins. See Figure 14 on page 178.

Ethernet SLA Operation Profile

An *operation profile* is a configuration entity that defines the following aspects of an operation:

- What packet types to send and in what quantities (probe and burst configuration)
- What statistics to measure, and how to aggregate them
- When to schedule the probes

An operation profile by itself does not cause any packets to be sent or statistics collected, but is used to create operation instances.

Ethernet SLA Operation

An *operation* is an instance of a given operation profile that is actively collecting performance data. Operation instances are created by associating an operation profile with a given source (an interface and MEP) and with a given destination (a MEP ID or MAC address). Operation instances exist for as long as the configuration is applied, and they run for an indefinite duration on an ongoing basis.

Ethernet SLA On-Demand Operation

An *on-demand operation* is a method of Ethernet SLA operation that can be run on an as-needed basis for a specific and finite period of time. This can be useful in situations such as when you are starting a new service or modifying the parameters for a service to verify the impact of the changes, or if you want to run a more detailed probe when a problem is detected by an ongoing scheduled operation.

On-demand operations do not use profiles and have a finite duration. The statistics that are collected are discarded after a finite time after the operation completes (two weeks), or when you manually clear them.

On-demand operations are not persistent so they are lost during certain events such as a card reload or Minimal Disruptive Restart (MDR).

Statistics Measurement and Ethernet SLA Operations Overview

Ethernet SLA statistics measurement for network performance is performed by sending packets and storing data metrics such as:

- Round-trip delay time—The time for a packet to travel from source to destination and back to source again.
- Round-trip jitter—The variance in round-trip delay time (latency).
- One-way delay and jitter—The router also supports measurement of one-way delay or jitter from source to destination, or from destination to source.

In addition to these metrics, the following statistics are also kept for SLA probe packets:

- Packet loss count
- Packet corruption event
- Out-of-order event

Counters for packet loss, corruption and out-of-order packets are kept for each bucket, and in each case, a percentage of the total number of samples for that bucket is reported (for example, 4% packet corruption). For delay and jitter statistics, the minimum, maximum, mean and standard deviation for the whole bucket are reported, as well as the individual samples or aggregated bins.

When aggregation is enabled using the **aggregate** command, bins are created to store a count of the samples that fall within a certain value range, which is set by the **width** keyword. Only a counter of the number of results that fall within the range for each bin is stored. This uses less memory than storing individual results. When aggregation is not used, each sample is stored separately, which can provide a more accurate statistics analysis for the operation, but it is highly memory-intensive due to the independent storage of each sample.

A bucket represents a time period during which statistics are collected. All the results received during that time period are recorded in the corresponding bucket. If aggregation is enabled, each bucket has its own set of bins and counters, and only results relating to the measurements initiated during the time period represented by the bucket are included in those counters.

By default, there is a separate bucket for each probe. The time period is determined by how long the probe lasts (configured by the **probe**, **send (SLA)**, and **schedule (SLA)** commands).You can modify the size of buckets so that you can have more buckets per probe or fewer buckets per probe (less buckets allows the results from multiple probes to be included in the same bucket). Changing the size of the buckets for a given metric clears all stored data for that metric. All existing buckets are deleted and new buckets are created.

Scheduled SLA operation profiles run indefinitely, according to a configured schedule, and the statistics that are collected are stored in a rolling buffer, where data in the oldest bucket is discarded when a new bucket needs to be recorded.

Configuration Overview of Scheduled Ethernet SLA Operations

When you configure a scheduled Ethernet SLA operation, you perform the following basic steps:

1. Configure global profiles to define how packets are sent in each probe, how the probes are scheduled, and how the results are stored.
2. Configure operations from a specific local MEP to a specific peer MEP using these profiles.

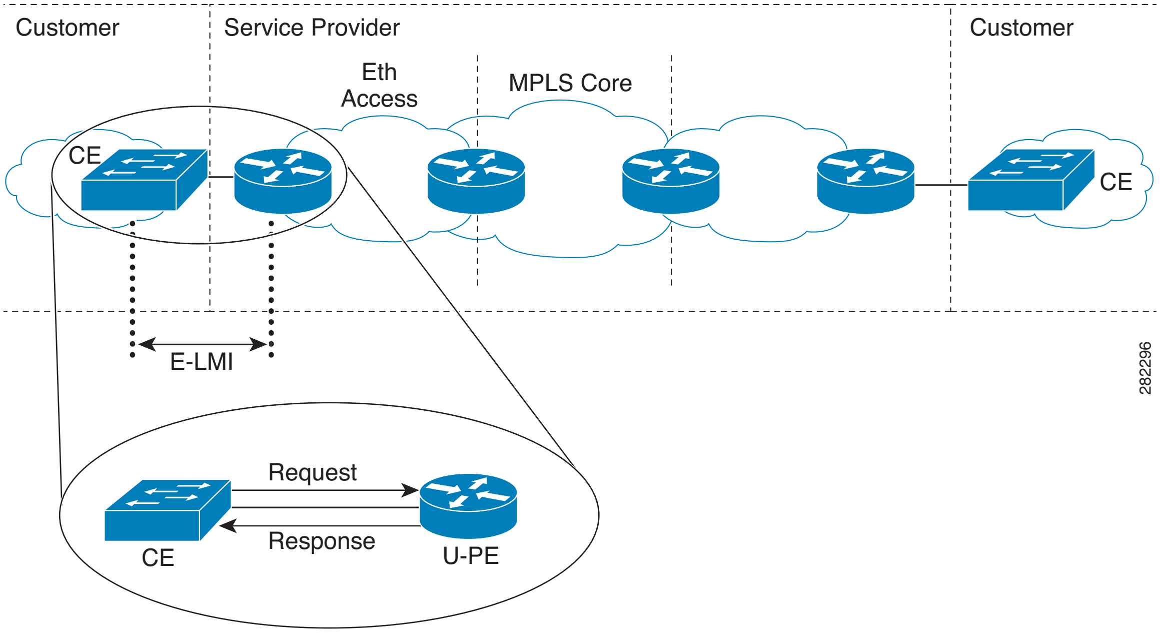
Note Certain Ethernet SLA configurations use large amounts of memory which can affect the performance of other features on the system. For more information, see the "Configuring Ethernet SLA" section on page 131.

Ethernet LMI

The Cisco ASR 9000 Series Router supports the Ethernet Local Management Interface (E-LMI) protocol as defined by the *Metro Ethernet Forum, Technical Specification MEF 16, Ethernet Local Management Interface (E-LMI), January 2006* standard.

E-LMI runs on the link between the customer-edge (CE) device and the provider-edge (PE) device, or User Network Interface (UNI), and provides a way for the CE device to auto-configure or monitor the services offered by the PE device (see Figure 13).

Figure 13 ***E-LMI Communication on CE-to-PE Link***



E-LMI is an asymmetric protocol whose basic operation involves the User-facing PE (uPE) device providing connectivity status and configuration parameters to the CE using STATUS messages in response to STATUS ENQUIRY messages sent by the CE to the uPE.

E-LMI Messaging

The E-LMI protocol as defined by the MEF 16 standard, defines the use of only two message types—STATUS ENQUIRY and STATUS.

These E-LMI messages consist of required and optional fields called information elements, and all information elements are associated with assigned identifiers. All messages contain the Protocol Version, Message Type, and Report Type information elements, followed by optional information elements and sub-information elements.

E-LMI messages are encapsulated in 46- to 1500-byte Ethernet frames, which are based on the IEEE 802.3 untagged MAC-frame format. E-LMI frames consist of the following fields:

- Destination address (6 bytes)—Uses a standard MAC address of 01:80:C2:00:00:07.
- Source address (6 bytes)—MAC address of the sending device or port.
- E-LMI Ethertype (2 bytes)—Uses 88-EE.
- E-LMI PDU (46–1500 bytes)—Data plus 0x00 padding as needed to fulfill minimum 46-byte length.
- CRC (4 bytes)—Cyclic Redundancy Check for error detection.

For more details about E-LMI messages and their supported information elements, refer to the *Metro Ethernet Forum, Technical Specification MEF 16, Ethernet Local Management Interface (E-LMI), January 2006.*

Cisco-Proprietary Remote UNI Details Information Element

The E-LMI MEF 16 specification does not define a way to send proprietary information.

To provide additional information within the E-LMI protocol, the Cisco IOS XR software implements a Cisco-proprietary information element called Remote UNI Details to send information to the CE about remote UNI names and states. This information element implements what is currently an unused identifier from the E-LMI MEF 16 specification.

To ensure compatibility for future implementations of E-LMI should this identifier ever be implemented in the standard protocol, or for another reason, you can disable transmission of the Remote UNI information element using the **extension remote-uni disable** command.

E-LMI Operation

The basic operation of E-LMI consists of a CE device sending periodic STATUS ENQUIRY messages to the PE device, followed by mandatory STATUS message responses by the PE device that contain the requested information. Sequence numbers are used to correlate STATUS ENQUIRY and STATUS messages between the CE and PE.

The CE sends the following two forms of STATUS ENQUIRY messages called Report Types:

- E-LMI Check—Verifies a Data Instance (DI) number with the PE to confirm that the CE has the latest E-LMI information.
- Full Status—Requests information from the PE about the UNI and all EVCs.

The CE device uses a polling timer to track sending of STATUS ENQUIRY messages, while the PE device can optionally use a Polling Verification Timer (PVT), which specifies the allowable time between transmission of the PE's STATUS message and receipt of a STATUS ENQUIRY from the CE device before recording an error.

In addition to the periodic STATUS ENQUIRY/STATUS message sequence for the exchange of E-LMI information, the PE device also can send asynchronous STATUS messages to the CE device to communicate changes in EVC status as soon as they occur and without any prompt by the CE device to send that information.

Both the CE and PE devices use a status counter (N393) to determine the local operational status of E-LMI by tracking consecutive errors received before declaring a change in E-LMI protocol status.

Supported E-LMI PE Functions on the Cisco ASR 9000 Series Router

The Cisco ASR 9000 Series Router serves as the PE device for E-LMI on a MEN, and supports the following PE functions:

- Supports the E-LMI protocol on Ethernet physical interfaces that are configured with Layer 2 subinterfaces as Ethernet Flow Points (EFPs), which serve as the EVCs about which the physical interface reports status to the CE. The Cisco IOS XR software does not support a specific manageability context for an Ethernet Virtual Connection (EVC).

Note For E-LMI on the Cisco ASR 9000 Series Router, the term EVC in this documentation refers to a Layer 2 subinterface/EFP.

- Provides the ability to configure the following E-LMI options defined in the MEF 16 specification:
  - T392 Polling Verification Timer (PVT)
  - N393 Status Counter
- Sends notification of the addition and deletion of an EVC.
- Sends notification of the availability (active) or unavailability (inactive, partially active) status of a configured EVC.
- Sends notification of the local UNI name.
- Sends notification of remote UNI names and states using the Cisco-proprietary Remote UNI Details information element, and the ability to disable the Cisco-proprietary Remote UNI information element.
- Sends information about UNI and EVC attributes to the CE (to allow the CE to auto-configure these attributes), including:
  - CE-VLAN to EVC Map
  - CE-VLAN Map Type (Bundling, All-to-one Bundling, Service Multiplexing)
  - Service Type (point-to-point or multipoint)
- Uses CFM Up MEPs to retrieve the EVC state, EVC Service Type, and remote UNI details.
- Provides the ability to retrieve the per-interface operational state of the protocol (including all the information currently being communicated by the protocol to the CE) using the command-line interface (CLI) or Extensible Markup Language (XML) interface.
- Supports up to 80 E-LMI sessions per linecard (one per physical interface).
- Supports up to 32000 EVCs total per linecard for all physical interfaces enabled for E-LMI.

Unsupported E-LMI Functions

These areas of E-LMI are not supported on the Cisco ASR 9000 Series Router:

- CE functions

Unidirectional Link Detection Protocol

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

UDLD Operation

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

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

This information can then be used to detect faults and miswiring conditions. The protocol operates an aging mechanism by means of which information from neighbors that is not periodically refreshed is eventually timed out. This mechanism can also be used for fault detection.

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

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

Types of Fault Detection

UDLD can detect these types of faults:

- **Transmit faults** — These are cases where there has been a failure in transmitting packets from the local port to the peer device, but packets continue to be received from the peer. These faults are caused by failure of the physical link (where notification at layer 1 of unidirectional link faults is not supported by the media) as well as packet path faults on the local or peer device.
- **Miswiring faults** — These are cases where the receiving and transmitting sides of a port on the local device are connected to different peer ports (on the same device or on different devices). This can occur when using unbundled fibers to connect fiber optic ports.
- **Loopback faults** — These are cases where the receiving and transmitting sides of a port are connected to each other, creating a loopback condition. This can be an intentional mode of operation, for certain types of testing, but UDLD must not be used in these cases.
- **Receive faults** — The protocol includes a heartbeat that is transmitted at a negotiated periodic interval to the peer device. Missed heartbeats can therefore be used to detect failures on the receiving side of the link (where they do not result in interface state changes). These could be caused by a unidirectional link with a failure only affecting the receiving side, or by a link which has developed a bidirectional fault. This detection depends on reliable, regular packet transmission by the peer device. For this reason, the UDLD protocol has two (configurable) modes of operation which determine the behavior on a heartbeat timeout. These modes are described in the section UDLD Modes of Operation, page 98.

UDLD Modes of Operation

UDLD can operate in these modes:

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

UDLD Aging Mechanism

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

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

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

State Machines

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

Main FSM

The Main FSM can be in one of these states:

- **Init**: Protocol is initializing.
- **UDLD inactive**: Port is down or UDLD is disabled.
- **Linkup**: Port is up and running, and UDLD is in the process of detecting a neighbor.
- **Detection**: A hello message from a new neighbor has been received and the Detection FSM is running to determine the status of the port.
- **Advertisement**: The Detection FSM has run and concluded that the port is operating correctly, periodic hellos will continue to be sent and hellos from neighbors monitored.
- **Port shutdown**: The Detection FSM detected a fault, or all neighbors were timed out in Aggressive mode, and the port has been disabled as a result.

Detection FSM

The Detection FSM can be in one of these states:

- **Unknown**: Detection has not yet been performed or UDLD has been disabled.
- **Unidirectional detected**: A unidirectional link condition has been detected because a neighbor does not see the local device, the port will be disabled.
- **Tx/Rx loop**: A loopback condition has been detected by receiving a TLV with the ports own identifiers, the port will be disabled.
- **Neighbor mismatch**: A miswiring condition has been detected in which a neighbor can identify other devices than those the local device can see and the port will be disabled.
- **Bidirectional detected**: UDLD hello messages are exchanged successfully in both directions, the port is operating correctly.

Unidirectional Link Routing

Unidirectional Link Routing (UDLR) feature allows a port to unidirectionally transmit or receive traffic. Therefore, instead of using two strands of fiber for a full-duplex Gigabit Ethernet or 10Gigabit Ethernet port, UDLR uses only one strand of fiber that either transmits or receives the one-way traffic depending on the configuration. This improves the effectiveness and also enables you to double the bandwidth with existing fiber infrastructure.

Cisco IOS XR Software supports Unidirectional Link Routing (UDLR) feature on these line cards:

- A9K- 24T-TR 24-port 10 Gigabit Ethernet line cards
- A9K- 24T-SE 24-port 10 Gigabit Ethernet line cards
- A9K- 36T-TR 36-port 10 Gigabit Ethernet line cards
- A9K- 36T-SE 36-port 10 Gigabit Ethernet line cards

UDLR is used for applications such as video streaming, where most of the traffic is sent as unacknowledged unidirectional video broadcast streams.

How to Configure Ethernet OAM

This section provides these configuration procedures:

- Configuring Ethernet Link OAM, page 100
- Configuring Ethernet CFM, page 108
- Configuring Ethernet SLA, page 131
- Configuring Ethernet LMI, page 141
- Configuring UDLD, page 159

Configuring Ethernet Link OAM

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

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

- Configuring an Ethernet OAM Profile, page 100
- Attaching an Ethernet OAM Profile to an Interface, page 105
- Configuring Ethernet OAM at an Interface and Overriding the Profile Configuration, page 106
- Verifying the Ethernet OAM Configuration, page 108

Configuring an Ethernet OAM Profile

Perform the following steps to configure an Ethernet OAM profile.

SUMMARY STEPS

1. **configure**
2. **ethernet oam profile** *profile-name*
3. **link-monitor**
4. **symbol-period window** *window*
5. **symbol-period threshold low** *threshold*
6. **frame window** *window*
7. **frame threshold low** *threshold*
8. **frame-period window** *window*
9. **frame-period threshold low** *threshold*
10. **frame-seconds window** *window*
11. **frame-seconds threshold low** *threshold*
12. **exit**
13. **mib-retrieval**
14. **connection timeout** *seconds*
15. **hello-interval** {**100ms** | **1s**}
16. **mode** {**active** | **passive**}
17. **require-remote mode** {**active** | **passive**}
18. **require-remote link-monitoring**
19. **require-remote mib-retrieval**
20. **action capabilities-conflict** {**disable** | **efd** | **error-disable-interface**}
21. **action critical-event** {**disable** | **error-disable-interface**}

22. **action discovery-timeout** {**disable** | **efd** | **error-disable-interface** }
23. **action dying-gasp** {**disable** | **error-disable-interface**}
24. **action high-threshold** {**error-disable-interface** | **log**}
25. **action remote-loopback disable**
26. **action session-down** {**disable** | **efd** | **error-disable-interface**}
27. **action session-up disable**
28. **action uni-directional link-fault** {**disable** | **efd** | **error-disable-interface**}
29. **action wiring-conflict** {**disable** | **efd** | **log**}
30. **uni-directional link-fault detection**
31. **commit**
32. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**ethernet oam profile** *profile-name* **Example:** `RP/0/RSP0/CPU0:router(config)# ethernet oam profile Profile_1`	Creates a new Ethernet Operations, Administration and Maintenance (OAM) profile and enters Ethernet OAM configuration mode.
Step 3	**link-monitor** **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# link-monitor`	Enters the Ethernet OAM link monitor configuration mode.
Step 4	**symbol-period window** *window* **Example:** `RP/0/RSP0/CPU0:router(config-eoam-lm)# symbol-period window 60000`	(Optional) Configures the window size (in milliseconds) for an Ethernet OAM symbol-period error event. The range is 1000 to 60000. The default value is 1000.
Step 5	**symbol-period threshold low** *threshold* **high** *threshold* **Example:** `RP/0/RSP0/CPU0:router(config-eoam-lm)# symbol-period threshold low 10000000 high 60000000`	(Optional) Configures the thresholds (in symbols) that trigger an Ethernet OAM symbol-period error event. The high threshold is optional and is configurable only in conjunction with the low threshold. The range is 0 to 60000000. The default low threshold is 1.

	Command or Action	Purpose

Step 6	**`frame window`** *`window`*
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# frame window 60`
		(Optional) Configures the frame window size (in milliseconds) of an OAM frame error event.
		The range is 1000 to 60000.
		The default value is 1000.

Step 7	**`frame threshold low`** *`threshold`* **`high`** *`threshold`*
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# frame threshold low 10000000 high 60000000`
		(Optional) Configures the thresholds (in symbols) that triggers an Ethernet OAM frame error event. The high threshold is optional and is configurable only in conjunction with the low threshold.
		The range is 0 to 60000000.
		The default low threshold is 1.

Step 8	**`frame-period window`** *`window`*
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# frame-period window 60000`
		(Optional) Configures the window size (in milliseconds) for an Ethernet OAM frame-period error event.
		The range is 100 to 60000.
		The default value is 1000.

Step 9	**`frame-period threshold low`** *`threshold`* **`high`** *`threshold`*
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# frame-period threshold low 100 high 1000000`
		(Optional) Configures the thresholds (in frames) that trigger an Ethernet OAM frame-period error event. The high threshold is optional and is configurable only in conjunction with the low threshold.
		The range is 0 to 1000000.
		The default low threshold is 60000.

Step 10	**`frame-seconds window`** *`window`*
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# frame-seconds window 900000`
		(Optional) Configures the window size (in milliseconds) for the OAM frame-seconds error event.
		The range is 10000 to 900000.
		The default value is 6000.

Step 11	**`frame-seconds threshold low`** *`threshold`* **`high`** *`threshold`*
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# frame-seconds threshold 3 threshold 900`
		(Optional) Configures the thresholds (in seconds) that trigger a frame-seconds error event. The high threshold value can be configured only in conjunction with the low threshold value.
		The range is 1 to 900
		The default value is 1.

Step 12	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam-lm)# exit`
		Exits back to Ethernet OAM mode.

Step 13	**`mib-retrieval`**
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# mib-retrieval`
		Enables MIB retrieval in an Ethernet OAM profile or on an Ethernet OAM interface.

	Command or Action	Purpose
Step 14	**connection timeout** *seconds* **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# connection timeout 30`	Configures the timeout value (in seconds) for an Ethernet OAM session. The range is 2 to 30. The default value is 5.

Step 15	**hello-interval** {**100ms**|**1s**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# hello-interval 100ms`
		Configures the time interval between hello packets for an Ethernet OAM session. The default is 1 second (**1s**).

Step 16	**mode** {**active**|**passive**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# mode passive`
		Configures the Ethernet OAM mode. The default is active.

Step 17	**require-remote mode** {**active**|**passive**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# require-remote mode active`
		Requires that active mode or passive mode is configured on the remote end before the OAM session becomes active.

Step 18	**require-remote link-monitoring** **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# require-remote link-monitoring`	Requires that link-monitoring is configured on the remote end before the OAM session becomes active.
Step 19	**require-remote mib-retrieval** **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# require-remote mib-retrieval`	Requires that MIB-retrieval is configured on the remote end before the OAM session becomes active.

Step 20	**action capabilities-conflict** {**disable** | **efd** | **error-disable-interface**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action capabilities-conflict efd`
		Specifies the action that is taken on an interface when a capabilities-conflict event occurs. The default action is to create a syslog entry.
		Note If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

Step 21	**action critical-event** {**disable** | **error-disable-interface**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action critical-event error-disable-interface`
		Specifies the action that is taken on an interface when a critical-event notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.
		Note If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

	Command or Action	Purpose

Step 22	**action discovery-timeout** {**disable** | **efd** | **error-disable-interface**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action discovery-timeout efd`
		Specifies the action that is taken on an interface when a connection timeout occurs. The default action is to create a syslog entry.
		Note If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

Step 23	**action dying-gasp** {**disable** | **error-disable-interface**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action dying-gasp error-disable-interface`
		Specifies the action that is taken on an interface when a dying-gasp notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.
		Note If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

Step 24	**action high-threshold** {**error-disable-interface** | **log**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action high-threshold error-disable-interface`
		Specifies the action that is taken on an interface when a high threshold is exceeded. The default is to take no action when a high threshold is exceeded.
		Note If you change the default, the **disable** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and take no action at the interface when the event occurs.

Step 25	**action remote-loopback disable** **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# action remote-loopback disable`	Specifies that no action is taken on an interface when a remote-loopback event occurs. The default action is to create a syslog entry. **Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

Step 26	**action session-down** {**disable** | **efd** | **error-disable-interface**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action session-down efd`
		Specifies the action that is taken on an interface when an Ethernet OAM session goes down.
		Note If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

Step 27	**action session-up disable** **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# action session-up disable`	Specifies that no action is taken on an interface when an Ethernet OAM session is established. The default action is to create a syslog entry. **Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.

	Command or Action	Purpose
Step 28	**action uni-directional link-fault** {**disable** | **efd** | **error-disable-interface**}	Specifies the action that is taken on an interface when a link-fault notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry. **Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. **Note** In Cisco IOS XR Release 4.0, this command replaces the **action link-fault** command.

Step 29	**action wiring-conflict** {**disable** | **efd** | **log**}
	Example:
	`RP/0/RSP0/CPU0:router(config-eoam)# action session-down efd`
		Specifies the action that is taken on an interface when a wiring-conflict event occurs. The default is to put the interface into error-disable state.
		Note If you change the default, the **error-disable-interface** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and put the interface into error-disable state when the event occurs.

Step 30	**uni-directional link-fault detection** **Example:** `RP/0/RSP0/CPU0:router(config-eoam)# uni-directional link-fault detection`	Enables detection of a local, unidirectional link fault and sends notification of that fault to an Ethernet OAM peer.
Step 31	**commit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# commit`	Saves the configuration changes to the running configuration file and remains within the configuration session.

Step 32	**end**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
		Ends the configuration session and exits to the EXEC mode.

Attaching an Ethernet OAM Profile to an Interface

Perform the following steps to attach an Ethernet OAM profile to an interface:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet oam**
4. **profile** *profile-name*
5. **commit**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/1/0/0`	Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. **Note** The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1.
Step 3	**ethernet oam** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ethernet oam`	Enables Ethernet OAM and enters interface Ethernet OAM configuration mode.
Step 4	**profile** *profile-name* **Example:** `RP/0/RSP0/CPU0:router(config-if-eoam)# profile Profile_1`	Attaches the specified Ethernet OAM profile (*profile-name*), and all of its configuration, to the interface.
Step 5	**commit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# commit`	Saves the configuration changes to the running configuration file and remains within the configuration session.

Step 6	**end**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
		Ends the configuration session and exits to the EXEC mode.

Configuring Ethernet OAM at an Interface and Overriding the Profile Configuration

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

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

To see all of the default Ethernet OAM configuration settings, see the "Verifying the Ethernet OAM Configuration" section on page 108.

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

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet oam**
4. *interface-Ethernet-OAM-command*
5. **commit**
6. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure terminal`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/1/0/0`
		Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.
		Note The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1.

Step 3	**`ethernet oam`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ethernet oam`
		Enables Ethernet OAM and enters interface Ethernet OAM configuration mode.

Step 4	*`interface-Ethernet-OAM-command`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-eoam)# action capabilities-conflict error-disable-interface`
		Configures a setting for an Ethernet OAM configuration command and overrides the setting for the profile configuration, where *interface-Ethernet-OAM-command* is one of the supported commands on the platform in interface Ethernet OAM configuration mode.

Step 5	**`commit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves the configuration changes to the running configuration file and remains within the configuration session.

Step 6	**`end`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
		Ends the configuration session and exits to the EXEC mode.

Verifying the Ethernet OAM Configuration

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

```
RP/0/RSP0/CPU0:router# show ethernet oam configuration
Thu Aug  5 22:07:06.870 DST
GigabitEthernet0/4/0/0:
  Hello interval:                                              1s
  Link monitoring enabled:                                      Y
  Remote loopback enabled:                                      N
  Mib retrieval enabled:                                        N
  Uni-directional link-fault detection enabled:                 N
  Configured mode:                                         Active
  Connection timeout:                                           5
  Symbol period window:                                         0
  Symbol period low threshold:                                  1
  Symbol period high threshold:                              None
  Frame window:                                              1000
  Frame low threshold:                                          1
  Frame high threshold:                                      None
  Frame period window:                                       1000
  Frame period low threshold:                                   1
  Frame period high threshold:                               None
  Frame seconds window:                                     60000
  Frame seconds low threshold:                                  1
  Frame seconds high threshold:                              None
  High threshold action:                                     None
  Link fault action:                                          Log
  Dying gasp action:                                          Log
  Critical event action:                                      Log
  Discovery timeout action:                                   Log
  Capabilities conflict action:                               Log
  Wiring conflict action:                           Error-Disable
  Session up action:                                          Log
  Session down action:                                        Log
  Remote loopback action:                                     Log
  Require remote mode:                                     Ignore
  Require remote MIB retrieval:                                 N
  Require remote loopback support:                              N
  Require remote link monitoring:                               N
```

Configuring Ethernet CFM

To configure Ethernet CFM, perform the following tasks:

- Configuring a CFM Maintenance Domain, page 109 (required)
- Configuring Services for a CFM Maintenance Domain, page 110 (required)
- Enabling and Configuring Continuity Check for a CFM Service, page 112 (optional)
- Configuring Automatic MIP Creation for a CFM Service, page 114 (optional)
- Configuring Cross-Check on a MEP for a CFM Service, page 116 (optional)
- Configuring Other Options for a CFM Service, page 118 (optional)
- Configuring CFM MEPs, page 120 (required)
- Configuring Y.1731 AIS, page 122 (optional)

- Configuring EFD for a CFM Service, page 126 (optional)
- Configuring Flexible VLAN Tagging for CFM, page 127 (optional)
- Verifying the CFM Configuration, page 129
- Troubleshooting Tips, page 129

Configuring a CFM Maintenance Domain

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **traceroute cache hold-time** *minutes* **size** *entries*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`ethernet cfm`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# ethernet cfm`
		Enters Ethernet Connectivity Fault Management (CFM) configuration mode.

Step 3	**`domain`** *`domain-name`* **`level`** *`level-value`* `[`**`id`** `[`**`null`**`] [`**`dns`** *`DNS-name`*`] [`**`mac`** *`H.H.H`*`] [`**`string`** *`string`*`] ]`
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1`
		Creates and names a container for all domain configurations and enters CFM domain configuration mode.
		The level must be specified.
		The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default.

	Command or Action	Purpose
Step 4	**traceroute cache hold-time** *minutes* **size** *entries* **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# traceroute cache hold-time 1 size 3000`	(Optional) Sets the maximum limit of traceroute cache entries or the maximum time limit to hold the traceroute cache entries. The default is 100 minutes and 100 entries.
Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# commit`	Saves configuration changes. • When you use the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Services for a CFM Maintenance Domain

You can configure up to 32000 CFM services for a maintenance domain.

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config)# ethernet cfm`	Enters Ethernet CFM configuration mode.
Step 3	**domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1`	Creates and names a container for all domain configurations at a specified maintenance level, and enters CFM domain configuration mode. The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default.

	Command or Action	Purpose
Step 4	**service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1`	Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created. The **id** sets the short MA name.
Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# commit`	Saves configuration changes. • When you use the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Enabling and Configuring Continuity Check for a CFM Service

The Cisco ASR 9000 Series Router supports Continuity Check as defined in the IEEE 802.1ag specification, and supports CCMs intervals of 100 ms and longer. The overall packet rates for CCM messages are up to 16000 CCMs-per-second sent, and up to 16000 CCMs-per-second received, per card.

Note If Ethernet SLA is configured, the overall combined packet rate for CCMs and SLA frames is 16000 frames-per-second in each direction, per card.

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]

4. **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **continuity-check interval** *time* [**loss-threshold** *threshold*]
6. **continuity-check archive hold-time** *minutes*
7. **continuity-check loss auto-traceroute**
8. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`ethernet cfm`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# ethernet cfm`
		Enters Ethernet Connectivity Fault Management (CFM) configuration mode.

Step 3	**domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1`	Creates and names a container for all domain configurations and enters the CFM domain configuration mode. The level must be specified. The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default.
Step 4	**service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1`	Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created. The **id** sets the short MA name.
Step 5	**continuity-check interval** *time* [**loss-threshold** *threshold*] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# continuity-check interval 100m loss-threshold 10`	(Optional) Enables Continuity Check and specifies the time interval at which CCMs are transmitted or to set the threshold limit for when a MEP is declared down.

	Command or Action	Purpose

Step 6	**`continuity-check archive hold-time`** *`minutes`*
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# continuity-check archive hold-time 100`
		(Optional) Configures how long information about peer MEPs is stored after they have timed out.

Step 7	**`continuity-check loss auto-traceroute`**
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# continuity-check loss auto-traceroute`
		(Optional) Configures automatic triggering of a traceroute when a MEP is declared down.

Step 8	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# commit`
		Saves configuration changes.
		• When you use the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Automatic MIP Creation for a CFM Service

For more information about the algorithm for creating MIPs, see the "MIP Creation" section on page 74.

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **mip auto-create** {**all** | **lower-mep-only**}

6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router# ethernet cfm`	Enters the Ethernet Connectivity Fault Management (CFM) configuration mode.
Step 3	**domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1`	Creates and names a container for all domain configurations and enters the CFM domain configuration mode. The level must be specified. The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default.
Step 4	**service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1`	Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created. The **id** sets the short MA name.

	Command or Action	Purpose

Step 5	`mip auto-create {all | lower-mep-only}`
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# mip auto-create all`
		(Optional) Enables the automatic creation of MIPs in a bridge domain or xconnect.

Step 6	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# commit`
		Saves configuration changes.
		• When you use the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Cross-Check on a MEP for a CFM Service

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps | xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **mep crosscheck**
6. **mep-id** *mep-id-number* [**mac-address** *mac-address*]
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router# ethernet cfm`	Enters the Ethernet Connectivity Fault Management (CFM) configuration mode.
Step 3	**domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1`	Creates and names a container for all domain configurations and enters the CFM domain configuration mode. The level must be specified. The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default.
Step 4	**service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1`	Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created. The **id** sets the short MA name.
Step 5	**mep crosscheck** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# mep crosscheck mep-id 10`	Enters CFM MEP crosscheck configuration mode.

	Command or Action	Purpose
Step 6	**mep-id** *mep-id-number* [**mac-address** *mac-address*] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-xcheck)# mep-id 10`	Enables cross-check on a MEP. **Note** Repeat this command for every MEP that you want included in the expected set of MEPs for cross-check.
Step 7	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-xcheck)# commit`	Saves configuration changes. • When you use the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Other Options for a CFM Service

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **maximum meps** *number*
6. **log** {**ais** | **continuity-check errors** | **continuity-check mep changes** | **crosscheck errors** | **efd**}
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router# ethernet cfm`	Enters the Ethernet Connectivity Fault Management (CFM) configuration mode.
Step 3	**domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1`	Creates and names a container for all domain configurations and enters the CFM domain configuration mode. The level must be specified. The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default.
Step 4	**service** *service-name* {**bridge group** *bridge-domain-group* **bridge-domain** *bridge-domain-name* | **down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**icc-based** *icc-string umc-string*] | [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1`	Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with a bridge domain or xconnect where MIPs and up MEPs will be created. The **id** sets the short MA name.
Step 5	**maximum-meps** *number* **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# maximum-meps 1000`	(Optional) Configures the maximum number (2 to 8190) of MEPs across the network, which limits the number of peer MEPs recorded in the database.

	Command or Action	Purpose

Step 6	**log** {**ais**|**continuity-check errors**|**continuity-check mep changes**|**crosscheck errors**|**efd**}
	
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# log continuity-check errors`
		(Optional) Enables logging of certain types of events.

Step 7	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# commit`
		Saves configuration changes.
		• When you use the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring CFM MEPs

When you configure CFM MEPs, consider the following guidelines:

- Up to 32000 local MEPs are supported per card.
- CFM maintenance points can be created on the following interface types:
  - All physical Ethernet interfaces (except for the RSP Management interfaces).
  - Ethernet bundle interfaces.
  - All physical and bundle Ethernet subinterfaces, providing the encapsulation is configured according to the following guidelines:

    Frames are only matched based on VLAN IDs and CoS bits.

    Frames are not matched using VLAN “any.”
  - Ethernet bundle member interfaces—Only down MEPs at level 0 can be created.
- CFM maintenance points can be created on both Layer 2 and Layer 3 interfaces. On L3 interfaces, only down MEPs can be created.

Restrictions

When you configure MEPs, consider the following restrictions:

- Maintenance points at level 0 are not supported on bundle interfaces.
- If a subinterface is configured that matches untagged Ethernet frames (for example, by configuring the **encapsulation default** command), then you can not create a down MEP on the underlying physical or bundle interface.
- Up MEPs are not supported on Layer 3 interfaces.

SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *interface-path-id***.***subinterface*
3. **ethernet cfm**
4. **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*
5. **cos** *cos*
6. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** {**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *interface-path-id***.***subinterface* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1`	Type of Ethernet interface on which you want to create a MEP. Enter **GigabitEthernet**, **TenGigE**, or **Bundle-Ether** and the physical interface or virtual interface followed by the subinterface path ID. Naming notation is *interface-path-id***.***subinterface*. The period in front of the subinterface value is required as part of the notation. For more information about the syntax for the router, use the question mark (**?**) online help function.
Step 3	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ethernet cfm`	Enters interface Ethernet CFM configuration mode.

	Command or Action	Purpose

Step 4	**`mep domain`** *`domain-name`* **`service`** *`service-name`* **`mep-id`** *`id-number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-cfm)# mep domain Dm1 service Sv1 mep-id 1`
		Creates a maintenance end point (MEP) on an interface and enters interface CFM MEP configuration mode.

Step 5	**`cos`** *`cos`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-cfm-mep)# cos 7`
		(Optional) Configures the class of service (CoS) (from 0 to 7) for all CFM packets generated by the MEP on an interface. If not configured, the CoS is inherited from the Ethernet interface.

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-cfm-mep)# commit`
		Saves configuration changes.
		• When you use the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Y.1731 AIS

This section has the following step procedures:

- Configuring AIS in a CFM Domain Service
- Configuring AIS on a CFM Interface

Configuring AIS in a CFM Domain Service

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

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *name* **level** *level*

4. **service** *name* **bridge group** *name* **bridge-domain** *name*
5. **ais transmission** [**interval** {**1s** | **1m**}][**cos** *cos*]
6. **log ais**
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config)# ethernet cfm`	Enters Ethernet CFM global configuration mode.
Step 3	**domain** *name* **level** *level* **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain D1 level 1`	Specifies the domain and domain level.
Step 4	**service** *name* **bridge group** *name* **bridge-domain** *name* **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S1 bridge group BG1 bridge-domain BD2`	Specifies the service, bridge group, and bridge domain.
Step 5	**ais transmission** [**interval** {**1s**|**1m**}][**cos** *cos*] **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7`	Configures Alarm Indication Signal (AIS) transmission for a Connectivity Fault Management (CFM) domain service.

	Command or Action	Purpose

Step 6	`log ais`
	Example:
	`RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# log ais`
		Configures AIS logging for a Connectivity Fault Management (CFM) domain service to indicate when AIS or LCK packets are received.

Step 7	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring AIS on a CFM Interface

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

SUMMARY STEPS

1. **configure**
2. **interface gigabitethernet** *interface-path-id*
3. **ethernet cfm**
4. **ais transmission up interval 1m cos** *cos*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface gigabitethernet** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# interface gigabitethernet 0/1/0/2`	Enters interface configuration mode.
Step 3	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config)# ethernet cfm`	Enters Ethernet CFM interface configuration mode.
Step 4	**ais transmission up interval 1m cos** *cos* **Example:** `RP/0/RSP0/CPU0:router(config-if-cfm)# ais transmission up interval 1m cos 7`	Configures Alarm Indication Signal (AIS) transmission on a Connectivity Fault Management (CFM) interface.
Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring EFD for a CFM Service

To configure EFD for a CFM service, complete the following steps.

Restrictions

EFD is not supported on up MEPs. It can only be configured on down MEPs, within a particular service.

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value*
4. **service** *service-name* **down-meps**
5. **efd**
6. **log efd**
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config)# ethernet cfm`	Enters CFM configuration mode.
Step 3	**domain** *domain-name* **level** *level-value* **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# domain D1 level 1`	Specifies or creates the CFM domain and enters CFM domain configuration mode.
Step 4	**service** *service-name* **down-meps** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S1 down-meps`	Specifies or creates the CFM service for down MEPS and enters CFM domain service configuration mode.
Step 5	**efd** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# efd`	Enables EFD on all down MEPs in the down MEPS service.

	Command or Action	Purpose
Step 6	**log efd** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# log efd`	(Optional) Enables logging of EFD state changes on an interface.
Step 7	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Verifying the EFD Configuration

The following example shows how to display all interfaces that are shut down because of Ethernet Fault Detection (EFD):

```
RP/0/RSP0/CPU0:router# show efd interfaces

Server VLAN MA
==============
Interface         Clients
-------------------------
GigE0/0/0/0.0     CFM
```

Configuring Flexible VLAN Tagging for CFM

Use the following procedure to set the number of tags in CFM packets from up MEPs to 1, in a CFM domain service.

SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *name* **level** *level*
4. **service** *name* **bridge group** *name* **bridge-domain** *name*

5. **tags** *number*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config)# ethernet cfm`	Enters Ethernet CFM global configuration mode.
Step 3	**domain** *name* **level** *level* **Example:** `RP/0/RSP0/CPU0:router(config-cfm)# domain D1 level 1`	Specifies the domain and domain level.
Step 4	**service** *name* **bridge group** *name* **bridge-domain** *name* **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S2 bridge group BG1 bridge-domain BD2`	Specifies the service, bridge group, and bridge domain.

	Command or Action	Purpose
Step 5	**tags** *number* **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# tags 1`	Specifies the number of tags in CFM packets from up MEPs. Currently, the only valid value is 1.
Step 6	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Verifying the CFM Configuration

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

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

Troubleshooting Tips

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

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

```
RP/0/RSP0/CPU0:router# ping ethernet cfm domain D1 service S1 mep-id 16 source
interface GigabitEthernet 0/0/0/0
```

```
Type escape sequence to abort.
Sending 5 CFM Loopbacks, timeout is 2 seconds -
Domain foo (level 2), Service foo
Source: MEP ID 1, interface GigabitEthernet0/0/0/0
Target: 0001.0002.0003 (MEP ID 16):
  Running (5s) ...
Success rate is 60.0 percent (3/5), round-trip min/avg/max = 1251/1349/1402 ms
Out-of-sequence: 0.0 percent (0/3)
Bad data: 0.0 percent (0/3)
Received packet rate: 1.4 pps
```

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

```
RP/0/RSP0/CPU0:router# traceroute ethernet cfm domain D1 service S1 mep-id 16 source
interface gigabitethernet 0/0/0/0

Traceroutes in domain D1 (level 4), service S1
Source: MEP-ID 1, interface GigabitEthernet0/0/0/0
================================================================================
Traceroute at 2009-05-18 12:09:10 to 0001.0203.0402,
TTL 64, Trans ID 2:

Hop Hostname/Last            Ingress MAC/name       Egress MAC/Name        Relay
--- ------------------------ ---------------------- ---------------------- -----
  1 ios                      0001.0203.0400 [Down]                         FDB
      0000-0001.0203.0400    Gi0/0/0/0
  2 abc                                             0001.0203.0401 [Ok]    FDB
      ios                                           Not present
  3 bcd                      0001.0203.0402 [Ok]                           Hit
      abc                    GigE0/0
Replies dropped: 0
```

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

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

Configuring Ethernet SLA

This section describes how to configure Ethernet SLA.

Ethernet SLA Configuration Guidelines

Caution Certain SLA configurations can use a large amount of memory which can affect the performance of other features on the router.

Before you configure Ethernet SLA, consider the following guidelines:

- Aggregation—Use of the **aggregate none** command significantly increases the amount of memory required because each individual measurement is recorded, rather than just counts for each aggregation bin. When you configure aggregation, consider that more bins will require more memory.
- Buckets archive—When you configure the **buckets archive** command, consider that the more history that is kept, the more memory will be used.
- Measuring two statistics (such as both delay and jitter) will use approximately twice as much memory as measuring one.
- Separate statistics are stored for one-way source-to-destination and destination-to-source measurements, which consumes twice as much memory as storing a single set of round-trip statistics.
- The Cisco ASR 9000 Series Router supports SLA packet intervals of 100 ms and longer. If Ethernet SLA is configured, the overall combined packet rate for CCMs and SLA frames is 16000 frames-per-second in each direction, per card.

The following procedure provides the steps to configure Ethernet Service Level Agreement (SLA) monitoring at Layer 2.

To configure SLA, perform the following tasks:

- Configuring an SLA Operation Profile, page 131
- Configuring SLA Probe Parameters in a Profile, page 132
- Configuring SLA Statistics Measurement in a Profile, page 134
- Configuring a Schedule for an SLA Operation Probe in a Profile, page 136
- Configuring an SLA Operation, page 138
- Configuring an On-Demand SLA Operation, page 139
- Verifying SLA Configuration, page 141

Configuring an SLA Operation Profile

To configure a profile, perform the following steps:

SUMMARY STEPS

1. **configure**
2. **ethernet sla**

3. **profile** *profile-name* **type** {**cfm-delay-measurement** | **cfm-loopback**}
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`ethernet sla`**
	Example:
	`RP/0/RSP0/CPU0:router# ethernet sla`
		Enters the SLA configuration mode.

Step 3	**`profile`** *`profile-name`* **`type`** {**`cfm-delay-measurement`** | **`cfm-loopback`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sla)# profile Prof1 type cfm-loopback`
		Creates an SLA operation profile and enters the SLA profile configuration mode.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sla)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring SLA Probe Parameters in a Profile

To configure SLA probe parameters in a profile, perform the following steps beginning in SLA profile configuration mode:

SUMMARY STEPS

1. **probe**
2. **send burst** {**every** *number* {**seconds** | **minutes** | **hours**}| **once**} **packet count** *packets* **interval** *number* {**seconds** | **milliseconds**}
   or
   send packet {**every** *number* {**milliseconds** | **seconds** | **minutes** | **hours**} | **once**}
3. **packet size** *bytes* [**test pattern** {**hex 0x***HHHHHHHH* | **pseudo-random**}]
4. **priority** *priority*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**probe** **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof)# probe`	Enters the SLA profile probe configuration mode.

Step 2	**send burst** {**every** *number* {**seconds** | **minutes** | **hours**} | **once**} **packet count** *packets* **interval** *number* {**seconds** | **milliseconds**}
	or
	send packet {**every** *number* {**milliseconds** | **seconds** | **minutes** | **hours**} | **once**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof-pb)# send burst every 60 seconds packet count 100 interval 100 milliseconds`
	or
	`RP/0/RSP0/CPU0:router(config-sla-prof-pb)# send burst once packet count 2 interval 1 second`
	or
	`RP/0/RSP0/CPU0:router(config-sla-prof-pb)# send packet every 100 milliseconds`
		Configures the number and timing of packets sent by a probe in an operations profile.

Step 3	**packet size** *bytes* [**test pattern** {**hex 0x***HHHHHHHH* | **pseudo-random**}] **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof-pb)# packet size 9000`	(CFM loopback probe types only) Configures the minimum size (in bytes) for outgoing probe packets, including padding when necessary. Use the test pattern keyword to specify a hexadecimal string to use as the padding characters, or a pseudo-random bit sequence. The default padding is 0's.

	Command or Action	Purpose

Step 4	**`priority`** *`priority`*
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof-pb)# priority 7`
		Configures the priority of outgoing SLA probe packets.

Step 5	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof-pb)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring SLA Statistics Measurement in a Profile

The Ethernet SLA feature supports measurement of one-way and two-way delay and jitter statistics.

Prerequisites

To configure one-way measurements, you must first configure the **profile (SLA)** command using the **type cfm-delay-measurement** form of the command.

To configure one-way delay measurements, be sure that the following clocking prerequisites are met:

- Frequency synchronization is configured globally in its default line timing mode (The **clock-interface timing mode** command is not configured.)
- The clock interface (Sync 0 / Sync 1) on the RSP is configured as a DTI port using the **port-parameters dti** command.
- Valid DTI input signals are available for the RSP clock-interface ports.
- Both Local and remote routers are using DTI timing input signals.

For more information about frequency synchronization configuration, see the “Configuring Ethernet Interfaces on the Cisco ASR 9000 Series Router”.

Restrictions

One-way delay and jitter measurements are not supported by cfm-loopback profile types.

To configure SLA statistics measurement in a profile, perform the following steps beginning in SLA profile configuration mode:

SUMMARY STEPS

1. **statistics measure** {**one-way-delay-ds** | **one-way-delay-sd** | **one-way-jitter-ds** | **one-way-jitter-sd** | **round-trip-delay** | **round-trip-jitter**}
2. **aggregate** {**bins** *count* **width** *width* | **none**}
3. **buckets size** *number* {**per-probe** | **probes**}
4. **buckets archive** *number*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**statistics measure** {**one-way-delay-ds** | **one-way-delay-sd** | **one-way-jitter-ds** | **one-way-jitter-sd** | **round-trip-delay** | **round-trip-jitter**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof)# statistics measure round-trip-delay`
		Enables the collection of SLA statistics, and enters SLA profile statistics configuration mode.

Step 2	**aggregate** {**bins** *count* **width** *width* | **none**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg) # aggregate bins 100 width 10000`
		Configures the size and number of bins into which to aggregate the results of statistics collection.

Step 3	**buckets size** *number* {**per-probe** | **probes}**
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg) # buckets size 100 per-probe`
		Configures the size of the buckets in which statistics are collected.

	Command or Action	Purpose
Step 4	**buckets archive** *number* **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg)# buckets archive 50`	Configures the number of buckets to store in memory.
Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring a Schedule for an SLA Operation Probe in a Profile

This section describes how to configure a schedule for an SLA operation probe on an ongoing basis within an SLA profile. For information about how to configure a schedule for a limited, on-demand SLA operation, see the "Configuring an On-Demand SLA Operation" section on page 139.

To configure a schedule for an SLA operation probe, perform the following steps beginning in SLA profile configuration mode:

SUMMARY STEPS

1. **schedule every week on** *day* [**at** *hh***:***mm*] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
   or
   schedule every day [**at** *hh***:***mm*] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
   or
   schedule every *number* {**hours** | **minutes**}[**first at** *hh***:***mm*[**.***ss*]] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
2. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**schedule every week on** *day* [**at** *hh***:***mm*] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
	or
	schedule every day [**at** *hh***:***mm*] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
	or
	schedule every *number* {**hours** | **minutes**}[**first at** *hh***:***mm*[**.***ss*]] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
	
	Example:
	`RP/0/RSP0/CPU0:router(config-sla-prof)# schedule every week on Monday at 23:30 for 1 hour`
	or
	`RP/0/RSP0/CPU0:router(config-sla-prof)# schedule every day at 11:30 for 5 minutes`
	or
	`RP/0/RSP0/CPU0:router(config-sla-prof)# schedule every 2 hours first at 13:45:01`
	or
	`RP/0/RSP0/CPU0:router(config-sla-prof)# schedule every 6 hours for 2 hours`
		Schedules an operation probe in a profile. A profile may contain only one schedule.

Step 2	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg) # commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring an SLA Operation

This section describes how to configure an ongoing SLA operation on a MEP using an SLA profile.

SUMMARY STEPS

1. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
2. **ethernet cfm**
3. **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*
4. **sla operation profile** *profile-name* **target** {**mep-id** *id* | **mac-address** *mac-address*}
5.
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# interface gigabitethernet 0/1/0/1`	Physical interface or virtual interface. **Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router. For more information about the syntax for the router, use the question mark (**?**) online help function.
Step 2	**ethernet cfm** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ethernet cfm`	Enters interface CFM configuration mode.
Step 3	**mep domain** *domain-name* **service** *service-name* **mep-id** *id-number* **Example:** `RP/0/RSP0/CPU0:router(config-if-cfm)# mep domain Dm1 service Sv1 mep-id 1`	Creates a MEP on an interface and enters interface CFM MEP configuration mode.

	Command or Action	Purpose

Step 4	**sla operation profile** *profile-name* **target** {**mep-id** *id* | **mac-address** *mac-address*}
	Example:
	`RP/0/RSP0/CPU0:router(config-if-cfm-mep)# sla operation profile Profile_1 target mac-address 01:23:45:67:89:ab`
		Creates an operation instance from a MEP to a specified destination.

Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-sla-prof-stat-cfg)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring an On-Demand SLA Operation

Beginning in Cisco IOS XR Release 4.0, the Cisco ASR 9000 Series Router supports configuration of on-demand SLA operations to run on an as-needed basis for a finite period of time.

This section includes the following topics:

- Configuration Guidelines, page 139
- Configuring an On-Demand Ethernet SLA Operation for CFM Delay Measurement, page 140
- Configuring an On-Demand Ethernet SLA Operation for CFM Loopback, page 141

Configuration Guidelines

When you configure on-demand SLA operations, consider the following guidelines:

- Each MEP supports up to 50 on-demand operations.
- Each card supports up to 250 on-demand operations.
- On-demand Ethernet SLA operations can be run in addition to any other ongoing scheduled SLA operations that you might have configured, and use similar amounts of CPU and router memory. When configuring an on-demand Ethernet SLA operation, you should consider your existing SLA operation configuration and the potential impact of additional packet processing to your normal operations.

- If you do not specify a schedule for the on-demand operation, the probe defaults to running one time beginning two seconds from the execution of the command, and runs for a ten-second duration.
- If you do not specify the statistics for the probe to measure, it defaults to measuring all statistics, inlcuding the following statistics by probe type:
  - CFM loopback—Two-way delay and jitter is measured by default.
  - CFM delay measurement—One-way delay and jitter in both directions, in addition to two-way delay and jitter is measured by default.
- The default operation mode is synchronous, where progress of the operation is reported to the console and the output of the statistics collection is displayed.

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

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

Command	Purpose
ethernet sla on-demand operation type cfm-delay-measurement probe [**priority** *number*] [**send** {**packet** {**once** | **every** *number* {**milliseconds** | **seconds** | **minutes** | **hours**}} | **burst** {**once** | **every** *number* {**seconds** | **minutes** | **hours**}} **packet count** *number* **interval** *number* {**milliseconds** | **seconds**}] **domain** *domain-name* **source interface** *type interface-path-id* **target** {**mac-address** *H.H.H.H* | **mep-id** *id-number*} [**statistics measure** {**one-way-delay-ds** | **one-way-delay-sd** | **one-way-jitter-ds** | **one-way-jitter-sd** | **round-trip-delay** | **round-trip-jitter**}][**aggregate** {**none** | **bins** *number* **width** *milliseconds*}] [**buckets** {**archive** *number* | **size** *number* {**per-probe** | **probes**}}] [**schedule** {**now** | **at** *hh*:*mm*[.*ss*] [*day* [*month* [*year*]]] | **in** *number* {**seconds** | **minutes** | **hours**}}[**for** *duration* {**seconds** | **minutes** | **hours**}][**repeat every** *number* {**seconds** | **minutes** | **hours**} **count** *probes*]] [**asynchronous**] **Example:** `RP/0/RSP0/CPU0:router# ethernet sla on-demand operation type cfm-delay-measurement probe domain D1 source interface TenGigE 0/6/1/0 target mep-id 100`	Configures an on-demand Ethernet SLA operation for CFM delay measurement. The example shows a minimum configuration, that specifies the local domain and source interface and target MEP, using the following defaults: • Send a burst once for a packet count of 10 and interval of 1 second (10-second probe). • Use default class of service (CoS) for the egress interface. • Measure all statistics, including both one-way and round-trip delay and jitter statistics. • Aggregate statistics into one bin. • Schedule now. • Display results on the console.

Configuring an On-Demand Ethernet SLA Operation for CFM Loopback

To configure an on-demand Ethernet SLA operation for CFM loopback, use the following command in privileged EXEC configuration mode:

Command	Purpose
ethernet sla on-demand operation type cfm-loopback probe [**packet size** *bytes* [**test pattern** {**hex 0x***HHHHHHHH* | **pseudo-random**}]] [**priority** *number*] [**send** {**packet** {**once** | **every** *number* {**milliseconds** | **seconds** | **minutes** | **hours**}} | **burst** {**once** | **every** *number* {**seconds** | **minutes** | **hours**}} **packet count** *number* **interval** *number* {**milliseconds** | **seconds**}] **domain** *domain-name* **source interface** *type interface-path-id* **target** {**mac-address** *H.H.H.H* | **mep-id** *id-number*} [**statistics measure** {**round-trip-delay** | **round-trip-jitter**}][**aggregate** {**none** | **bins** *number* **width** *milliseconds*}][**buckets** {**archive** *number* | **size** *number* {**per-probe** | **probes**}}] [**schedule** {**now** | **at** *hh*:*mm*[.*ss*] [*day* [*month* [*year*]]] | **in** *number* {**seconds** | **minutes** | **hours**}}[**for** *duration* {**seconds** | **minutes** | **hours**}][**repeat every** *number* {**seconds** | **minutes** | **hours**} **count** *probes*]] [**asynchronous**] **Example:** `RP/0/RSP0/CPU0:router# ethernet sla on-demand operation type cfm-loopback probe packet size 1500 domain D1 source interface TenGigE 0/6/1/0 target mep-id 100`	Configures an on-demand Ethernet SLA operation for CFM loopback. The example shows a minimum configuration, but specifies the option of a minimum packet size, and specifies the local domain and source interface and target MEP, using the following defaults: • Send a burst once for a packet count of 10 and interval of 1 second (10-second probe). • Use default test pattern of 0's for padding. • Use default class of service (CoS) for the egress interface. • Measure all statistics. • Aggregate statistics into one bin. • Schedule now. • Display results on the console.

Verifying SLA Configuration

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

Command	Purpose
show ethernet sla configuration-errors [**domain** *domain-name*] [**interface** *interface-path-id*] [**profile** *profile-name*]	Displays information about errors that are preventing configured SLA operations from becoming active, as well as any warnings that have occurred.
show ethernet sla operations [**detail**] [**domain** *domain-name*] [**interface** *interface-path-id*] [**profile** *profile-name*]	Displays information about configured SLA operations.

Configuring Ethernet LMI

To configure Ethernet LMI, complete the following tasks:

- Prerequisites for Configuring E-LMI, page 142
- Restrictions for Configuring E-LMI, page 142
- Creating EVCs for E-LMI, page 142 (required)

- Configuring Ethernet CFM for E-LMI, page 146 (required)
- Configuring UNI Names on the Physical Interface, page 148 (optional)
- Enabling E-LMI on the Physical Interface, page 149 (required)
- Configuring the Polling Verification Timer, page 151 (optional)
- Configuring the Status Counter, page 152 (optional)
- Disabling Syslog Messages for E-LMI Errors or Events, page 154 (optional)
- Disabling Use of the Cisco-Proprietary Remote UNI Details Information Element, page 155 (optional)
- Verifying the Ethernet LMI Configuration, page 157
- Troubleshooting Tips for E-LMI Configuration, page 157

Prerequisites for Configuring E-LMI

Before you configure E-LMI on the Cisco ASR 9000 Series Router, be sure that you complete the following requirements:

- Identify the local and remote UNIs in your network where you want to run E-LMI, and define a naming convention for them.
- Enable E-LMI on the corresponding CE interface link on a device that supports E-LMI CE operation, such as the Cisco Catalyst 3750 Metro Series Switches.

Restrictions for Configuring E-LMI

When configuring E-LMI, consider the following restrictions:

- E-LMI is not supported on subinterfaces or bundle interfaces. E-LMI is configurable on Ethernet physical interfaces only.

Creating EVCs for E-LMI

EVCs for E-LMI on the Cisco ASR 9000 Series Router are established by first configuring EFPs (Layer 2 subinterfaces) on the local UNI physical Ethernet interface link to the CE where E-LMI will be running, and also on the remote UNI link. Then, the EFPs need to be assigned to an L2VPN bridge domain to create the EVC.

To create EVCs, complete the following tasks:

- Configuring EFPs, page 142 (required)
- Configuring a Bridge Group and Assigning EFPs to a Bridge Domain, page 145 (required)

Configuring EFPs

This section describes the basic configuration of an EFP. For more information about configuration of other supported Layer 2 services, see the *Cisco ASR 9000 Series Aggregation Services Routers L2VPN and Ethernet Services Configuration Guide*.

To configure an EFP, complete the following tasks:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id***.***subinterface* **l2transport**
3. **encapsulation dot1q** *vlan-id* [**, untagged** | **,** *vlan-id* | **–***vlan-id*] [**exact** | **ingress source-mac** *mac-address* | **second-dot1q** *vlan-id*]
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id***.***subinterface* **l2transport** **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/0/0/0.0 l2transport`	Creates a VLAN subinterface in Layer 2 transport mode and enters Layer 2 subinterface configuration mode.
Step 3	**encapsulation dot1q** *vlan-id* [**,** **untagged** | **,** *vlan-id* | **-***vlan-id*] [**exact** | **ingress source-mac** *mac-address* | **second-dot1q** *vlan-id*] Example: `RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 1-20`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance.
Step 4	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring a Bridge Group and Assigning EFPs to a Bridge Domain

To configure a bridge group and assign EFPs to a bridge domain to create an EVC, complete the following steps:

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *name*
4. **bridge-domain** *name*
5. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id***.***subinterface*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# l2vpn`
		Enters L2VPN configuration mode.

Step 3	**`bridge group`** *`bridge-group-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group BG1`
		Creates a bridge group and enters L2VPN bridge group configuration mode.

Step 4	**`bridge-domain`** *`bridge-domain-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain BD1`
		Creates a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

	Command or Action	Purpose
Step 5	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id.subinterface* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/0/0/0.0`	Associates the EFP (EVC) with the specified bridge domain and enters L2VPN bridge group bridge domain attachment circuit configuration mode, where *interface-path-id* is specified as the *rack/slot/module/port* location of the interface and **.***subinterface* is the subinterface number. Repeat this step for as many EFPs (EVCs) as you want to associate with the bridge domain.

Step 6	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Ethernet CFM for E-LMI

The Cisco ASR 9000 Series Router uses Ethernet CFM to monitor EVC status for E-LMI. To use CFM for E-LMI, a CFM maintenance domain and service must be configured on the router and the EFPs must be configured as CFM Up MEPs.

To configure Ethernet CFM for E-LMI, complete the following tasks:

- Configuring Ethernet CFM, page 108 (required)
- Configuring EFPs as CFM Up MEPs, page 147 (required)

Configuring Ethernet CFM

The minimum configuration to support E-LMI using Ethernet CFM is to configure a CFM maintenance domain and service on the router. Other CFM options can also be configured.

For more information about the tasks to configure Ethernet CFM, see the "Configuring Ethernet CFM" section on page 108.

Configuring EFPs as CFM Up MEPs

This section describes the minimum tasks required to configure EFPs as CFM MEPs. For more information about configuring CFM MEPs, see the "Configuring CFM MEPs" section on page 120.

To configure EFPs as CFM MEPs, complete the following tasks for each E-LMI EFP:

SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id***.***subinterface*
3. **ethernet cfm**
4. **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`interface gigabitethernet interface-path-id.subinterface l2transport`
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/0/0/0.0 l2transport`
		Enters Layer 2 subinterface configuration mode for the EFP.

Step 3	`ethernet cfm`
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# ethernet cfm`
		Enters Ethernet CFM interface configuration mode.

	Command or Action	Purpose
Step 4	**mep domain** *domain-name* **service** *service-name* **mep-id** *id-number* **Example:** `RP/0/RSP0/CPU0:router(config-if-cfm)# mep domain GLOBAL service CustomerA mep-id 22`	Creates a MEP on an interface and enters interface CFM MEP configuration mode.
Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-if-cfm-mep)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring UNI Names on the Physical Interface

It is recommended that you configure UNI names on the physical interface links to both the local and remote UNIs to aid in management for the E-LMI protocol. To configure UNI names, complete the following tasks on the physical interface links to both the local and remote UNIs:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet uni id** *name*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/0/0/0`
		Enters interface configuration mode for the physical interface.

Step 3	**`ethernet uni id`** *`name`*
	`Example:`
	`RP/0/RSP0/CPU0:router(config-if)# ethernet uni id PE1-CustA-Slot0-Port0`
		Specifies a name (up to 64 characters) for the Ethernet UNI interface link.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Enabling E-LMI on the Physical Interface

The Cisco ASR 9000 Series Router supports the E-LMI protocol only on physical Ethernet interfaces. To enable E-LMI, complete the following tasks on the physical Ethernet interface link to the local UNI:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*

3. **ethernet lmi**
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# interface gigabitethernet 0/0/0/0`
		Enters interface configuration mode for the physical interface.

Step 3	**`ethernet lmi`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ethernet lmi`
		Enables Ethernet Local Managment Interface operation on an interface and enters interface Ethernet LMI configuration mode.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-lmi)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Polling Verification Timer

The MEF T392 Polling Verification Timer (PVT) specifies the allowable time between transmission of a STATUS message and receipt of a STATUS ENQUIRY from the UNI-C before recording an error. The default value is 15 seconds.

To modify the default value or disable the PVT altogether, complete the following tasks:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet lmi**
4. **polling-verification-timer** {*interval* | **disable**}
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# interface gigabitethernet 0/0/0/0`	Enters interface configuration mode for the physical interface.
Step 3	**ethernet lmi** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ethernet lmi`	Enables Ethernet Local Managment Interface operation on an interface and enters interface Ethernet LMI configuration mode.

	Command or Action	Purpose

Step 4	`polling-verification-timer {interval | disable}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-lmi)# polling-verification-timer 30`
		Sets or disables the MEF T392 Polling Verification Timer for E-LMI operation, which specifies the allowable time (in seconds) between transmission of a STATUS message and receipt of a STATUS ENQUIRY from the UNI-C before recording an error. The default is 15.

Step 5	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-lmi)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Status Counter

The MEF N393 Status Counter value is used to determine E-LMI operational status by tracking receipt of consecutive good packets or successive expiration of the PVT on packets. The default counter is four, which means that while the E-LMI protocol is in Down state, four good packets must be received consecutively to change the protocol state to Up, or while the E-LMI protocol is in Up state, four consecutive PVT expirations must occur before the state of the E-LMI protocol is changed to Down on the interface.

To modify the status counter default value, complete the following tasks:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet lmi**
4. **status-counter** *threshold*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# interface gigabitethernet 0/0/0/0`	Enters interface configuration mode for the physical interface.
Step 3	**ethernet lmi** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ethernet lmi`	Enables Ethernet Local Managment Interface operation on an interface and enters interface Ethernet LMI configuration mode.
Step 4	**status-counter** *threshold* **Example:** `RP/0/RSP0/CPU0:router(config-if-lmi)# status-counter 5`	Sets the MEF N393 Status Counter value that is used to determine E-LMI operational status by tracking receipt of consecutive good and bad packets from a peer. The default is 4.
Step 5	**end** or **commit** **Example:** `RP/0/RSP0/CPU0:router(config-if-lmi)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling Syslog Messages for E-LMI Errors or Events

The E-LMI protocol tracks certain errors and events whose counts can be displayed using the **show ethernet lmi interfaces** command.

To disable syslog messages for E-LMI errors or events, complete the following tasks:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet lmi**
4. **log** {**errors** | **events**} **disable**
5. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`interface [GigabitEthernet | TenGigE] interface-path-id`
	
	Example:
	`RP/0/RSP0/CPU0:router# interface gigabitethernet 0/0/0/0`
		Enters interface configuration mode for the physical interface.

Step 3	`ethernet lmi`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ethernet lmi`
		Enables Ethernet Local Managment Interface operation on an interface and enters interface Ethernet LMI configuration mode.

	Command or Action	Purpose

Step 4	**`log`** `{`**`errors`** `|` **`events`**`}` **`disable`**
	
	`Example:`
	`RP/0/RSP0/CPU0:router(config-if-lmi)# log events disable`
		Turns off syslog messages for E-LMI errors or events.

Step 5	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if-lmi)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling Use of the Cisco-Proprietary Remote UNI Details Information Element

To provide additional information within the E-LMI protocol, the Cisco IOS XR software implements a Cisco-proprietary information element called Remote UNI Details to send information to the CE about remote UNI names and states. This information element implements what is currently an unused identifier from the E-LMI MEF 16 specification.

To disable use of the Remote UNI Details information element, complete the following tasks:

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet lmi**
4. **extension remote-uni disable**
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# interface gigabitethernet 0/0/0/0`
		Enters interface configuration mode for the physical interface.

Step 3	**`ethernet lmi`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ethernet lmi`
		Enables Ethernet Local Managment Interface operation on an interface and enters interface Ethernet LMI configuration mode.

Step 4	**`extension remote-uni disable`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-lmi)# extension remote-uni disable`
		Disables transmission of the Cisco-proprietary Remote UNI Details information element in E-LMI STATUS messages.

Step 5	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-lmi)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Verifying the Ethernet LMI Configuration

Use the **show ethernet lmi interfaces detail** command to display the values for the Ethernet LMI configuration for a particular interface, or for all interfaces. The following example shows sample output for the command:

```
RP/0/RSP0/CPU0:router# show ethernet lmi interfaces detail
Interface: GigabitEthernet0/0/0/0
  Ether LMI Link Status: Up
  UNI Id: PE1-CustA-Slot0-Port0
  Line Protocol State: Up
  MTU: 1514 (1 PDU reqd. for full report)
  CE-VLAN/EVC Map Type: Bundling (1 EVC)
  Configuration: Status counter 4, Polling Verification Timer 15 seconds
  Last Data Instance Sent: 0
  Last Sequence Numbers: Sent 0, Received 0

  Reliability Errors:
    Status Enq Timeouts                  0 Invalid Sequence Number             0
    Invalid Report Type                  0

  Protocol Errors:
    Malformed PDUs                       0 Invalid Procotol Version            0
    Invalid Message Type                 0 Out of Sequence IE                  0
    Duplicated IE                        0 Mandatory IE Missing                0
    Invalid Mandatory IE                 0 Invalid non-Mandatory IE            0
    Unrecognized IE                      0 Unexpected IE                       0

  Full Status Enq Received     never       Full Status Sent         never
  PDU Received                 never       PDU Sent                 never
  LMI Link Status Changed   00:00:03 ago   Last Protocol Error      never
  Counters cleared             never

  Sub-interface: GigabitEthernet0/0/0/0.0
    VLANs: 1-20
    EVC Status: Active
    EVC Type: Point-to-Point
    OAM Protocol: CFM
      CFM Domain: Global (level 5)
      CFM Service: CustomerA
    Remote UNI Count: Configured = 1, Active = 1

    Remote UNI Id                                                  Status
    -------------                                                  ------
    PE1-CustA-Slot0-Port1                                          Up
```

Troubleshooting Tips for E-LMI Configuration

This section describes some basic information for troubleshooting your E-LMI configuration in the following topics:

- Ethernet LMI Link Status Troubleshooting, page 158
- Ethernet LMI Line Protocol State Troubleshooting, page 158
- Ethernet LMI Error Counter Troubleshooting, page 159
- Ethernet LMI Remote UNI Troubleshooting, page 159

Ethernet LMI Link Status Troubleshooting

The E-LMI protocol operational status is reported in the “Ether LMI Link Status” or “ELMI state” fields in the output of forms of the **show ethernet lmi interfaces** command. To investigate a link status other than “Up,” consider the following guidelines:

- Unknown (PVT disabled)—Indicates that the Polling Verification Timer has been configured as disabled, so no status information can be provided. To see an “Up” or “Down” status, you must enable the PVT. For more information, see the “Configuring the Polling Verification Timer” section on page 151.
- Down—The E-LMI link status can be Down for the following reasons:
  - The PVT has timed out the number of times specified by the **status-counter** command. This indicates that STATUS ENQUIRY messages have not been received from the CE device. This can be for the following reasons:

    — The CE device is not connected to the PE device. Check that the CE device is connected to the interface on which E-LMI is enabled on the PE device.

    — The CE device is not sending Status Enquiries. Check that E-LMI is enabled on the CE interface which is connected to the PE device.

    — Protocol errors are causing the PVT to expire. The PVT is only reset when a valid (unerrored) STATUS ENQUIRY message is received.
  - The Line Protocol State is “Down” or “Admin Down.”
  - The protocol has not yet started on the interface because it does not have useful information to provide, such as the UNI Id or details about EVCs. This is a symptom of provisioning misconfiguration.

Note If the protocol is started, then E-LMI still responds to STATUS ENQUIRY messages when it is in “Down” state.

Ethernet LMI Line Protocol State Troubleshooting

The E-LMI line protocol state is reported in the “Line Protocol State” or “LineP State” fields in the output of forms of the **show ethernet lmi interfaces** command. The line protocol state is the state of the E-LMI protocol on the physical interface.

To investigate a line protocol state other than Up, consider the following guidelines:

- Admin-Down—The interface is configured with the **shutdown** command. Use the **no shutdown** command to bring the interface up.
- Down—Indicates a fault on the interface. Run the **show interfaces** command to display both the interface state and the interface line protocol state for more information, and take the following actions to investigate further:
  - If both states are Down, this suggests a physical problem with the link (for example, the cable is not plugged into either the PE or CE device).
  - If the interface state is Up but the line protocol state is Down, this suggests that an OAM protocol has brought the line protocol state down due to a fault. Use the **show efd interface** command for more information.

Ethernet LMI Error Counter Troubleshooting

The **show ethernet lmi interfaces** command displays two sections of error counters:

- Reliability Errors—Can indicate that messages are being lost between the PE and CE devices. The timers in the last block of the output should indicate that messages are being sent and received by the PE device.
- Protocol Errors—Indicates that the CE device is sending packets to the PE device, but the PE does not understand those packets. This suggests an incorrect configuration of the E-LMI protocol on the CE side, or corruption of the packets on the path between the CE and PE. E-LMI packets have a strictly defined structure in the MEF 16 standard, and any deviation from that results in a protocol error. The PE will not respond to any packets that are malformed and result in a protocol error.

Immediately after configuring E-LMI, all of the error counters should be zero, with the possible exception of the Status Enq Timeouts counter. The Status Enq Timeouts counter can be non-zero if the E-LMI protocol was started on the PE interface before being started on the corresponding CE interface. However, once the protocol is started on both devices, this counter should stop increasing.

If the Status Enq Timeouts counter is non-zero and is increasing, this indicates that enquiries are not being received from the CE device. This can be due to the following conditions:

- The CE device is not connected or not sending STATUS ENQUIRY messages. For more information, see also the "Ethernet LMI Link Status Troubleshooting" section on page 158.
- The Polling Timer on the CE device is configured to a value greater than the PVT on the PE device. Verify that the value of the **polling-verification-timer** command on the PE device is larger than the value of the CE's Polling Timer.

For more information, see also the documentation for the show ethernet lmi interfaces command in the *Cisco ASR 9000 Aggregation Services Router Interfaces and Hardware Component Command Reference*.

Ethernet LMI Remote UNI Troubleshooting

Information about the Remote UNIs is reported in the output of the **show ethernet lmi interfaces detail** command. The Remote UNI ID field displays the name of the UNI as configured by the **ethernet uni id** command, or it displays the CFM MEP ID of the UNI when the UNI name has not been configured.

If the Remote UNI is missing from the table altogether, this is can be due to the following conditions:

- The remote UNI's EFP is missing from the bridge-domain in L2VPN configuration. Use the **show ethernet cfm configuration-errors** command to verify the configuration.
- A CFM MEP has not been configured on the remote UNI's EFP.

Configuring UDLD

UDLD is configured for each interface. The interface must be a physical ethernet interface.

Perform these steps to configure UDLD protocol on an interface.

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet udld**

4. **mode {normal | aggressive}**
5. **message-time**
6. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/1/0/0`
		Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.
		Note The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1.

Step 3	**`ethernet udld`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ethernet udld`
		Enables ethernet UDLD function and enters interface Ethernet UDLD configuration mode.

Step 4	**`mode {normal |aggressive}`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-udld)# mode normal`
		(Optional) Specifies the mode of operation for UDLD. The options are normal and aggressive.

Step 5	**`message-time`** *`[7-90]`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-udld)# message-time 70`
		(Optional) Specifies the message time (in seconds) to use for the UDLD protocol. The value ranges between 7 to 90 seconds.

Step 6	**`logging disable`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-udld)# loggig disable`
		(Optional) This command suppresses the operational UDLD syslog messages.

Step 7	**`end`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-udld)# end`
		Ends the configuration session and exits to the EXEC mode.

Configuration Examples for Ethernet OAM

This section provides the following configuration examples:

Configuration Examples for EOAM Interfaces

This section provides the following configuration examples:

Configuring an Ethernet OAM Profile Globally: Example

The following example shows how to configure an Ethernet OAM profile globally:

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

Configuring Ethernet OAM Features on an Individual Interface: Example

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

```
configure terminal
 interface TenGigE 0/1/0/0
  ethernet oam
   link-monitor
    symbol-period window 60000
    symbol-period threshold low 10000000 high 60000000
    frame window 60
    frame threshold low 10000000 high 60000000
    frame-period window 60000
    frame-period threshold low 100 high 12000000
    frame-seconds window 900000
    frame-seconds threshold 3 threshold 900
    exit
   mib-retrieval
   connection timeout 30
   require-remote mode active
   require-remote link-monitoring
   require-remote mib-retrieval
   action link-fault  error-disable-interface
   action dying-gasp error-disable-interface
   action critical-event error-disable-interface
   action discovery-timeout error-disable-interface
   action session-down error-disable-interface
   action capabilities-conflict error-disable-interface
   action wiring-conflict error-disable-interface
   action remote-loopback error-disable-interface
   commit
```

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

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

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

configure terminal
 interface TenGigE 0/1/0/0
  ethernet oam
   profile Profile_1
    mode active
    action dying-gasp log
    action critical-event log
    action discovery-timeout log
    action session-up log
    action session-down log
    action capabilities-conflict log
    action wiring-conflict log
    action remote-loopback log
    action uni-directional link-fault log
    uni-directional link-fault detection
    commit
```

Configuring a Remote Loopback on an Ethernet OAM Peer: Example

The following example shows how to configure a remote loopback on an Ethernet OAM peer:

```
configure terminal
 interface gigabitethernet 0/1/5/6
  ethernet oam
   profile Profile_1
   remote-loopback
```

The following example shows how to start a remote loopback on a configured Ethernet OAM interface:

```
ethernet oam loopback enable TenGigE 0/6/1/0
```

Clearing Ethernet OAM Statistics on an Interface: Example

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

```
RP/0/RP0/CPU0:router# clear ethernet oam statistics interface gigabitethernet 0/1/5/1
```

Enabling SNMP Server Traps on a Router: Example

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

```
configure terminal
  ethernet oam profile Profile_1
  snmp-server traps ethernet oam events
```

Configuration Examples for Ethernet CFM

This section includes the following examples:

- Ethernet CFM Domain Configuration: Example, page 164
- Ethernet CFM Service Configuration: Example, page 164
- Flexible Tagging for an Ethernet CFM Service Configuration: Example, page 164
- Continuity Check for an Ethernet CFM Service Configuration: Example, page 164
- MIP Creation for an Ethernet CFM Service Configuration: Example, page 164
- Cross-check for an Ethernet CFM Service Configuration: Example, page 164
- Other Ethernet CFM Service Parameter Configuration: Example, page 165
- MEP Configuration: Example, page 165
- Ethernet CFM Show Command: Examples, page 165
- AIS for CFM Configuration: Examples, page 168
- AIS for CFM Show Commands: Examples, page 169
- EFD Configuration: Examples, page 173
- Displaying EFD Information: Examples, page 173

Ethernet CFM Domain Configuration: Example

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

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

Ethernet CFM Service Configuration: Example

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

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

Flexible Tagging for an Ethernet CFM Service Configuration: Example

The following example shows how to set the number of tags in CFM packets from up MEPs in a CFM domain service:

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

Continuity Check for an Ethernet CFM Service Configuration: Example

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

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

MIP Creation for an Ethernet CFM Service Configuration: Example

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

```
   mip auto-create all
   commit
```

Cross-check for an Ethernet CFM Service Configuration: Example

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

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

Other Ethernet CFM Service Parameter Configuration: Example

The following example shows how to configure other Ethernet CFM service options:

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

MEP Configuration: Example

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

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

Ethernet CFM Show Command: Examples

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

Example 1

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm local maintenance-points

Domain/Level         Service             Interface         Type   ID   MAC
-------------------- ------------------- ----------------- ------ ---- --------
fig/5                bay                 Gi0/10/0/12.23456 Dn MEP    2 44:55:66
fig/5                bay                 Gi0/0/1/0.1       MIP         55:66:77
fred/3               barney              Gi0/1/0/0.1       Up MEP    5 66:77:88!
```

Example 2

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm configuration-errors

Domain fig (level 5), Service bay
 * MIP creation configured using bridge-domain blort, but bridge-domain blort does not
exist.
 * An Up MEP is configured for this domain on interface GigabitEthernet0/1/2/3.234 and an
Up MEP is also configured for domain blort, which is at the same level (5).
 * A MEP is configured on interface GigabitEthernet0/3/2/1.1 for this domain/service,
which has CC interval 100ms, but the lowest interval supported on that interface is 1s
```

Example 3

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm local meps

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

Domain foo (level 6), Service bar
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ------------------------ --- -------- -- ------- ---
  100 Gi1/1/0/1.234 (Up)       Up     0/0   N  A       L7

Domain fred (level 5), Service barney
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ------------------------ --- -------- -- ------- ---
    2 Gi0/1/0/0.234 (Up)       Up     3/2   Y  RPC     L6
```

Example 4

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm peer meps

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

Domain fred (level 7), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
St    ID MAC address    Port    Up/Downtime   CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- --------- ------ ----- -----
 >     1 0011.2233.4455 Up      00:00:01         1234      0     0     0
R>     4 4455.6677.8899 Up      1d 03:04         3456      0   234     0
L      2 1122.3344.5566 Up      3w 1d 6h         3254      0     0  3254
C      2 7788.9900.1122 Test    00:13            2345      6    20  2345
X      3 2233.4455.6677 Up      00:23              30      0     0    30
I      3 3344.5566.7788 Down    00:34           12345      0   300  1234
V      3 8899.0011.2233 Blocked 00:35              45      0     0    45
 T     5 5566.7788.9900         00:56              20      0     0     0
M      6                                            0      0     0     0
U>     7 6677.8899.0011 Up      00:02             456      0     0     0

Domain fred (level 7), Service fig
Down MEP on GigabitEthernet0/10/0/12.123, MEP-ID 3
================================================================================
St    ID MAC address    Port    Up/Downtime  CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- -------- ------ ----- -----
 >     1 9900.1122.3344 Up      03:45           4321      0     0     0
```

Example 5

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm peer meps detail
Domain dom3 (level 5), Service ser3
Down MEP on GigabitEthernet0/0/0/0 MEP-ID 1
================================================================================
Peer MEP-ID 10, MAC 0001.0203.0403
   CFM state: Wrong level, for 00:01:34
   Port state: Up
   CCM defects detected:    V - Wrong Level
   CCMs received: 5
```

```
     Out-of-sequence:             0
     Remote Defect received:      5
     Wrong Level:                 0
     Cross-connect (wrong MAID):  0
     Wrong Interval:              5
     Loop (our MAC received):     0
     Config (our ID received):    0
Last CCM received 00:00:06 ago:
     Level: 4, Version: 0, Interval: 1min
     Sequence number: 5, MEP-ID: 10
     MAID: String: dom3, String: ser3
     Port status: Up, Interface status: Up


Domain dom4 (level 2), Service ser4
Down MEP on GigabitEthernet0/0/0/0 MEP-ID 1
================================================================================
Peer MEP-ID 20, MAC 0001.0203.0402
   CFM state: Ok, for 00:00:04
   Port state: Up
   CCMs received: 7
     Out-of-sequence:             1
     Remote Defect received:      0
     Wrong Level:                 0
     Cross-connect (wrong MAID):  0
     Wrong Interval:              0
     Loop (our MAC received):     0
     Config (our ID received):    0
Last CCM received 00:00:04 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 20
     MAID: String: dom4, String: ser4
     Chassis ID: Local: ios; Management address: 'Not specified'
     Port status: Up, Interface status: Up

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


Domain dom5 (level 2), Service ser5
Up MEP on Standby Bundle-Ether 1 MEP-ID 1
================================================================================
Peer MEP-ID 600, MAC 0001.0203.0401
   CFM state: Ok (Standby), for 00:00:08, RDI received
   Port state: Down
   CCM defects detected:    Defects below ignored on local standby MEP
                            I - Wrong Interval
                            R - Remote Defect received
   CCMs received: 5
     Out-of-sequence:             0
     Remote Defect received:   5
     Wrong Level:                 0
     Cross-connect W(wrong MAID): 0
     Wrong Interval:              5
     Loop (our MAC received):     0
     Config (our ID received):    0
   Last CCM received 00:00:08 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 600
     MAID: DNS-like: dom5, String: ser5
     Chassis ID: Local: ios; Management address: 'Not specified'
     Port status: Up, Interface status: Down

Peer MEP-ID 601, MAC 0001.0203.0402
   CFM state: Timed Out (Standby), for 00:15:14, RDI received
   Port state: Down
   CCM defects detected:    Defects below ignored on local standby MEP
                            I - Wrong Interval
                            R - Remote Defect received
                            T - Timed Out
                            P - Peer port down
   CCMs received: 2
     Out-of-sequence:             0
     Remote Defect received:      2
     Wrong Level:                 0
     Cross-connect (wrong MAID):  0
     Wrong Interval:              2
     Loop (our MAC received):     0
     Config (our ID received):    0
   Last CCM received 00:15:49 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 600
     MAID: DNS-like: dom5, String: ser5
     Chassis ID: Local: ios; Management address: 'Not specified'
     Port status: Up, Interface status: Down
```

AIS for CFM Configuration: Examples

Example 1

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

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# ethernet cfm
RP/0/RSP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S1 bridge group BG1 bridge-domain BD2
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7
```

Example 2

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

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# ethernet cfm
RP/0/RSP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S2 bridge group BG1 bridge-domain BD2
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# log ais
```

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

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/2
RP/0/RSP0/CPU0:router(config-if)# ethernet cfm
RP/0/0RP0RSP0/CPU0:router(config-if-cfm)# ais transmission up interval 1m cos 7
```

AIS for CFM Show Commands: Examples

This section includes the following examples:

- show ethernet cfm interfaces ais Command: Example, page 170
- show ethernet cfm local meps Command: Examples, page 170

show ethernet cfm interfaces ais Command: Example

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm interfaces ais

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

                               Trigger                 Transmission
                         AIS  ---------    Via    ---------------------------
Interface (State)        Dir  L Defects  Levels   L Int Last started  Packets
------------------------ ---  - -------  -------  - --- ------------ --------
Gi0/1/0/0.234 (Up)       Dn   5 RPC      6        7 1s  01:32:56 ago     5576
Gi0/1/0/0.567 (Up)       Up   0 M        2,3      5 1s  00:16:23 ago      983
Gi0/1/0/1.1 (Dn)         Up     D                 7 60s 01:02:44 ago     3764
Gi0/1/0/2 (Up)           Dn   0 RX       1!
```

show ethernet cfm local meps Command: Examples

Example 1: Default

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm local meps

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

Domain foo (level 6), Service bar
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ------------------------ --- -------- -- ------- ---
  100 Gi1/1/0/1.234 (Up)       Up     0/0   N  A       7

Domain fred (level 5), Service barney
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ------------------------ --- -------- -- ------- ---
    2 Gi0/1/0/0.234 (Up)       Up     3/2   Y  RPC     6
```

Example 2: Domain Service

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

```
RP/0/RSP0RP0/CPU0:router# show ethernet cfm local meps domain foo service bar detail

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

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

```
Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

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

Example 3: Verbose

The following example shows how to display verbose statistics for MEPs in a domain service:

Note The Discarded CCMs field is not displayed when the number is zero (0). It is unusual for the count of discarded CCMs to be any thing other than zero, since CCMs are only discarded when the limit on the number of peer MEPs is reached.

```
RP/0/RSP0RP0/CPU0:router# show ethernet cfm local meps domain foo service bar verbose

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

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

  Packet        Sent      Received
  ------  ----------  -----------------------------------------------------
  CCM              0             0  (out of seq: 0)
  LBM              0             0
  LBR              0             0  (out of seq: 0, with bad data: 0)
  AIS           5576             0
  LCK              -             0

Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

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

```
Packet        Sent      Received
------  ----------  ----------------------------------------------------------
CCM          12345         67890  (out of seq: 6, discarded: 10)
LBM              5             0
LBR              0             5  (out of seq: 0, with bad data: 0)
AIS              0         46910
LCK              -             0
```

Example 4: Detail

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm local meps detail

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

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

Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

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

EFD Configuration: Examples

The following example shows how to enable EFD:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# ethernet cfm
RP/0/RSP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S1 down-meps
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# efd
```

The following example shows how to enable EFD logging:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# ethernet cfm
RP/0/RSP0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/RSP0/CPU0:router(config-cfm-dmn)# service S1 down-meps
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# log efd
```

Displaying EFD Information: Examples

The following examples show how to display information about EFD:

- show efd interfaces Command: Example, page 174
- show ethernet cfm local meps detail Command: Example, page 174

show efd interfaces Command: Example

The following example shows how to display all interfaces that are shut down in response to an EFD action:

```
RP/0/RSP0/CPU0:router# show efd interfaces

Server VLAN MA
==============
Interface         Clients
-------------------------
GigE0/0/0/0.0     CFM
```

show ethernet cfm local meps detail Command: Example

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

```
RP/0/RSP0/CPU0:router# show ethernet cfm local meps detail

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 2 missing, 0 unexpected

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

Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 0 missing, 0 unexpected

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

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

Configuration Examples for Ethernet SLA

This section includes the following examples:

- Ethernet SLA Profile Type Configuration: Examples, page 175
- Ethernet SLA Probe Configuration: Examples, page 175
- Profile Statistics Measurement Configuration: Examples, page 176

- Scheduled SLA Operation Probe Configuration: Examples, page 177
- Ethernet SLA Operation Probe Scheduling and Aggregation Configuration: Example, page 177
- Ongoing Ethernet SLA Operation Configuration: Example, page 178
- On-Demand Ethernet SLA Operation Basic Configuration: Examples, page 179
- Ethernet SLA Show Commands: Examples, page 179

Ethernet SLA Profile Type Configuration: Examples

The following examples show how to configure the different profile types supported by Ethernet SLA.

Example 1

This example configures a profile named "Prof1" for CFM loopback measurements:

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
   commit
```

Example 2

This example configures a profile named "Prof1" for CFM delay measurements. Setting this type allows you to configure the probe to measure additional one-way delay and jitter statistics:

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
   commit
```

Ethernet SLA Probe Configuration: Examples

The following examples show how to configure some of the packet options for an Ethernet CFM loopback probe.

Example 1

This example shows how to configure sending a group of 100 packets in 100 ms intervals and repeat that burst every 60 seconds. Packets are padded to a size of 9000 bytes as needed using a hexadecimal test pattern of "abcdabcd," and with a class of service value of 7:

Note The total length of a burst (packet count multiplied by the interval) must not exceed 1 minute.

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
   probe
    send burst every 60 seconds packet count 100 interval 100 milliseconds
    packet size 9000 test pattern hex 0xabcdabcd
    priority 7
    commit
```

Example 2

This example has the same characteristics as the configuration in Example 1, but sends a single burst of 50 packets, one second apart:

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
   probe
    send burst once packet count 50 interval 1 second
    packet size 9000 test pattern hex 0xabcdabcd
    priority 7
    commit
```

Example 3

This example shows how to configure a continuous stream of packets at 100 ms intervals for the duration of the probe. Packets are padded to a size of 9000 bytes as needed using a pseudo-random test pattern, and with a class of service value of 7:

```
configure
 ethernet sla
  profile Prof1 type cfm-loopback
   probe
    send burst every 60 seconds packet count 600 interval 100 milliseconds
    packet size 9000 test pattern pseudo-random
    priority 7
    commit
```

Profile Statistics Measurement Configuration: Examples

The following examples show how to configure the different types of statistics measurement.

Example 1

This example shows the two available types of statistics that can be measured by a CFM loopback SLA profile type:

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
  statistics measure round-trip-delay
  statistics measure round-trip-jitter
   commit
```

Example 2

This example shows how to configure measurement of round-trip delay and one-way jitter (from destination to source) for a CFM delay measurement SLA profile type:

Note The CFM delay measurement profile type supports measurement of all round-trip and one-way delay and jitter statistics.

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  statistics measure round-trip-delay
  statistics measure one-way-jitter-ds
   commit
```

Scheduled SLA Operation Probe Configuration: Examples

The following examples show how to configure different schedules for an SLA operation probe.

Example 1

This example shows how to configure a probe to run hourly for a specified duration:

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  schedule every 1 hours for 15 minutes
  commit
```

Example 2

This example shows how to configure a probe to run daily for a specified period of time:

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  schedule every day at 11:30 for 5 minutes
  commit
```

Example 3

This example shows how to configure a probe to run weekly beginning at a specified time and for a specified duration:

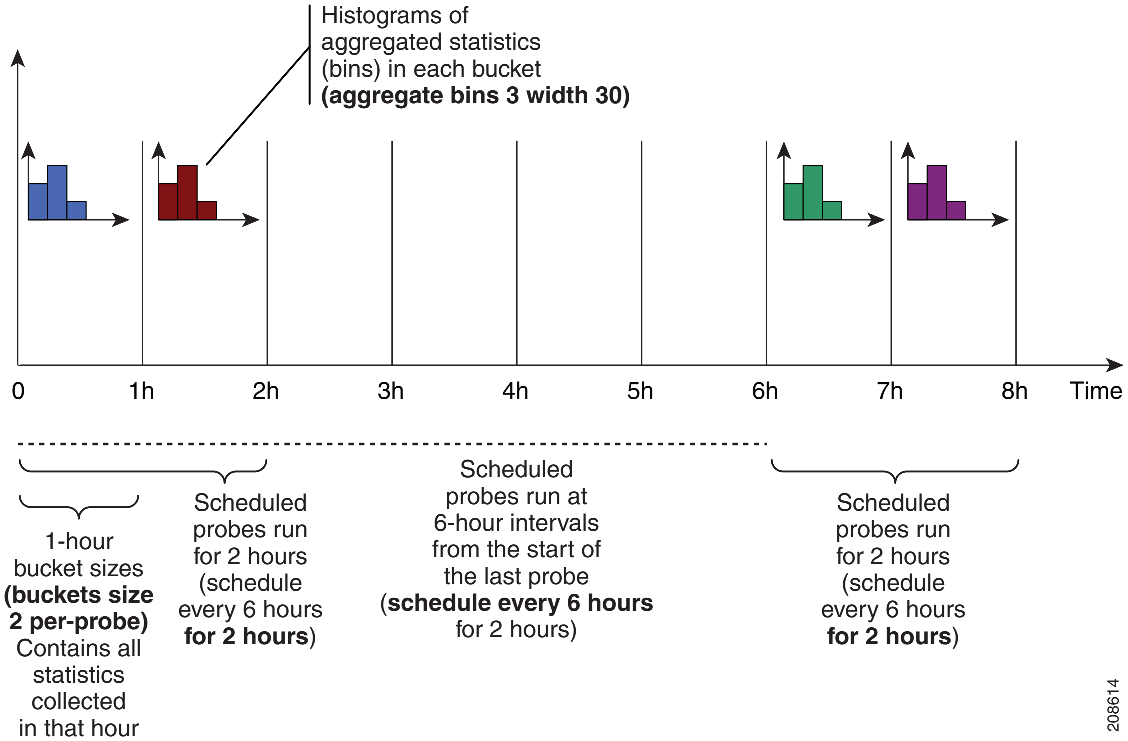
```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  schedule every week on Monday at 23:30 for 1 hour
  commit
```

Ethernet SLA Operation Probe Scheduling and Aggregation Configuration: Example

Figure 14 shows a more comprehensive example of how some of the probe scheduling and measurement configuration works using aggregation. The following configuration supports some of the concepts shown in the figure:

```
configure
 ethernet sla profile Prof1 type cfm-loopback
  probe
   send packet every 60 seconds
   schedule every 6 hours for 2 hours
   statistics measure round-trip-delay
    aggregate bins 3 width 30
    buckets size 2 per-probe
    buckets archive 4
    commit
```

Figure 14 SLA Probe Scheduling Operation With Bin Aggregation


Histograms of aggregated statistics (bins) in each bucket **(aggregate bins 3 width 30)**
0
1h
2h
3h
4h
5h
6h
7h
8h
Time
1-hour bucket sizes **(buckets size 2 per-probe)** Contains all statistics collected in that hour
Scheduled probes run for 2 hours (schedule every 6 hours **for 2 hours**)
Scheduled probes run at 6-hour intervals from the start of the last probe (**schedule every 6 hours** for 2 hours)
Scheduled probes run for 2 hours (schedule every 6 hours **for 2 hours**)
208614

This example schedules a probe with the following characteristics:

- Sends packets 60 seconds apart (for a 2-hour probe, this results in sending 120 individual packets).
- Probe runs every 6 hours for 2 hours duration.
- Collects data into 2 buckets for every probe, so each bucket covers 1 hour of the 2-hour probe duration.
- Aggregates statistics within the buckets into 3 bins each in the following ranges:
  - Bin 1 contains samples in the range 0 to < 30 ms.
  - Bin 2 contains samples in the range 30 ms to < 60 ms.
  - Bin 3 contains samples in the range 60 ms or greater (unbounded).
- The last 4 buckets are saved in memory.

Ongoing Ethernet SLA Operation Configuration: Example

The following example shows how to configure an ongoing Ethernet SLA operation on a MEP:

```
interface gigabitethernet 0/1/0/1
 ethernet cfm
 mep domain Dm1 service Sv1 mep-id 1
 sla operation profile Profile_1 target mac-address 01:23:45:67:89:ab s
 commit
 end
```

On-Demand Ethernet SLA Operation Basic Configuration: Examples

The following examples show how to configure on-demand Ethernet SLA operations.

Example 1

The following example shows how to configure a basic on-demand Ethernet SLA operation for a CFM loopback probe that by default will measure round-trip delay and round-trip jitter for a one-time, 10-second operation to the target MEP:

```
RP/0/RSP0/CPU0:router# ethernet sla on-demand operation type cfm-loopback probe domain D1
source interface TenGigE 0/6/1/0 target mep-id 1
```

Example 2

The following example shows how to configure a basic on-demand Ethernet SLA operation for a CFM delay measurement probe that by default will measure one-way delay and jitter in both directions, as well as round-trip delay and round-trip jitter for a one-time, 10-second operation to the target MEP:

```
RP/0/RSP0/CPU0:router# ethernet sla on-demand operation type cfm-delay-measurement probe
domain D1 source interface TenGigE 0/6/1/0 target mep-id 1
```

Ethernet SLA Show Commands: Examples

The following examples show how to display information about configured SLA operations:

show ethernet sla operations Command: Example 1

```
RP/0/RSP0/CPU0:router# show ethernet sla operations interface gigabitethernet 0/1/0/1.1

Interface GigabitEthernet0/1/0/1.1
Domain mydom Service myser to 00AB.CDEF.1234
-----------------------------------------------------------------------------
Profile 'business-gold'
Probe type CFM-delay-measurement:
    bursts sent every 1min, each of 20 packets sent every 100ms
    packets padded to 1500 bytes with zeroes
    packets use priority value of 7
Measures RTT: 5 bins 20ms wide; 2 buckets/ probe; 75/100 archived
Measures Jitter (interval 1): 3 bins 40ms wide; 2 buckets/probe; 50 archived
Scheduled to run every Sunday at 4am for 2 hours:
    last run at 04:00 25/05/2008
```

show ethernet sla configuration-errors Command: Example 2

```
RP/0/RSP0/CPU0:router# show ethernet sla configuration-errors

Errors:
-------
  Profile 'gold' is not defined but is used on Gi0/0/0/0.0
  Profile 'red' defines a test-pattern, which is not supported by the type
```

The following examples show how to display the contents of buckets containing SLA metrics collected by probes:

show ethernet sla statistics current Command: Example 3

```
RP/0/RSP0/CPU0:router# show ethernet sla statistics current interface GigabitEthernet
0/0/0/0.0

Interface GigabitEthernet 0/0/0/0.0
Domain mydom Service myser to 00AB.CDEF.1234
```

```
=============================================================================
Profile 'business-gold', packet type 'cfm-loopback'
Scheduled to run every Sunday at 4am for 2 hours

Round Trip Delay
~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: 13ms; Max: 154ms; Mean: 28ms; StdDev: 11ms

Round Trip Jitter
~~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: -5ms; Max: 8ms; Mean: 0ms; StdDev: 3.6ms

Bucket started at 05:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: 0; Max: 4; Mean: 1.4; StdDev: 1
```

show ethernet sla statistics history detail Command: Example 4

```
RP/0/RSP0/CPU0:router# show ethernet sla history detail GigabitEthernet 0/0/0/0.0

Interface GigabitEthernet 0/0/0/0.0
Domain mydom Service myser to 00AB.CDEF.1234
===============================================================================
Profile 'business-gold', packet type 'cfm-loopback'
Scheduled to run every Sunday at 4am for 2 hours

Round Trip Delay
~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: 13ms, occurred at 04:43:29 on Sun 22 Aug 2010 UTC
    Max: 154ms, occurred at 05:10:32 on Sun 22 Aug 2010 UTC
    Mean: 28ms; StdDev: 11ms

    Results suspect as more than 10 seconds time drift detected
    Results suspect as scheduling latency prevented some packets being sent

    Samples:
    Time sent       Result  Notes
    ------------  --------  ----------
    04:00:01.324      23ms
    04:00:01.425      36ms
    04:00:01.525         -  Timed Out
    ...

Round Trip Jitter
~~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008, lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: -5ms; Max: 10ms; Mean: 0ms; StdDev: 3.6ms

    Samples:
```

```
    Time sent       Result  Notes
    ------------  --------  ----------
    04:00:01.324         -
    04:00:01.425      13ms
    04:00:01.525         -  Timed out
    ...
```

show ethernet sla statistics history detail on-demand: Example 5

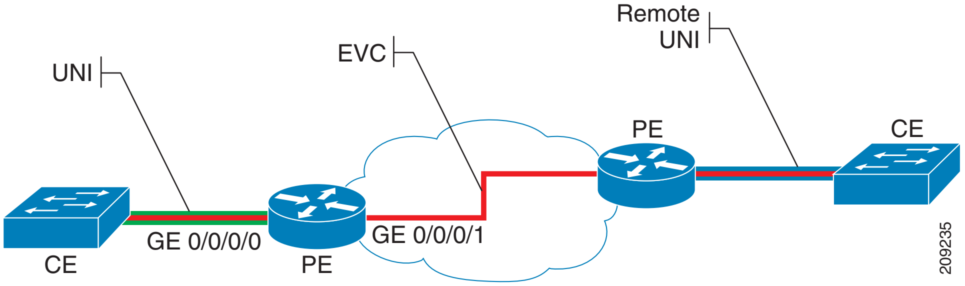
The following example shows how to display statistics for all full buckets for on-demand operations in detail:

```
RP/0/RSP0/CPU0/router #show ethernet sla statistics history detail on-demand

Interface GigabitEthernet0/0/0/0.1
Domain mydom Service myser to 0123.4567.890A
=============================================================================
On-demand operation ID #1, packet type 'cfm-delay-measurement'
Started at 15:38 on 06 July 2010 UTC, runs every 1 hour for 1 hour

Round Trip Delay
~~~~~~~~~~~~~~~~
1 bucket per probe

Bucket started at 15:38 on Tue 06 Jul 2010 UTC, lasting 1 hour:
    Pkts sent: 1200; Lost: 4 (0%); Corrupt: 600 (50%); Misordered: 0 (0%)
    Min: 13ms, occurred at 15:43:29 on Tue 06 Jul 2010 UTC
    Max: 154ms, occurred at 16:15:34 on Tue 06 Jul 2010 UTC
    Mean: 28ms; StdDev: 11ms

    Bins:
    Range              Samples    Cum. Count      Mean
    ------------  ------------  ------------  --------
     0 - 20 ms       194 (16%)     194 (16%)      17ms
    20 - 40 ms       735 (61%)     929 (77%)      27ms
    40 - 60 ms       212 (18%)    1141 (95%)      45ms
    > 60    ms        55  (5%)    1196            70ms

Bucket started at 16:38 on Tue 01 Jul 2008 UTC, lasting 1 hour:
    Pkts sent: 3600; Lost: 12 (0%); Corrupt: 1800 (50%); Misordered: 0 (0%)
    Min: 19ms, occurred at 17:04:08 on Tue 06 Jul 2010 UTC
    Max: 70ms, occurred at 16:38:00 on Tue 06 Jul 2010 UTC
    Mean: 28ms; StdDev: 11ms

    Bins:
    Range              Samples    Cum. Count      Mean
    ------------  ------------  ------------  --------
     0 - 20 ms       194 (16%)     194 (16%)      19ms
    20 - 40 ms       735 (61%)     929 (77%)      27ms
    40 - 60 ms       212 (18%)    1141 (95%)      45ms
    > 60    ms        55  (5%)    1196            64ms
```

Configuration Example for Ethernet LMI

Figure 15 shows a basic E-LMI network environment with a local UNI defined on a Cisco ASR 9000 Series Router functioning as the PE using Gigabit Ethernet interface 0/0/0/0, and connectivity to a remote UNI over Gigabit Ethernet interface 0/0/0/1.

Figure 15 Basic E-LMI UNI and Remote UNI Diagram



The following configuration provides a basic E-LMI configuration for the environment shown in Figure 15, for the Cisco ASR 9000 Series Router as the PE device on the local UNI with physical Gigabit Ethernet interfaces 0/0/0/0 and 0/0/0/1:

```
RP/0/RSP0/CPU0:router# configure
!
! Configure the Local UNI EFPs
!
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/0/0/0.0 l2transport
RP/0/RSP0/CPU0:router(config-subif)# #encapsulation dot1q 1-20
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/0/0/1.1 l2transport
RP/0/RSP0/CPU0:router(config-subif)# #encapsulation dot1q 1-20
RP/0/RSP0/CPU0:router(config-subif)# exit
!
! Create the EVC
!
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group BG1
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain BD1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet0/0/0/0.0
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet0/0/0/1.1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# exit
RP/0/RSP0/CPU0:router(config-l2vpn)# exit
!
! Configure Ethernet CFM
!
RP/0/RSP0/CPU0:router(config)# ethernet cfm
RP/0/RSP0/CPU0:router(config-cfm)# domain GLOBAL level 5
RP/0/RSP0/CPU0:router(config-cfm-dmn)# service CustomerA bridge group BG1 bridge-domain
BD1
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# continuity-check interval 100ms
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# mep crosscheck mep-id 22
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# mep crosscheck mep-id 11
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# exit
RP/0/RSP0/CPU0:router(config-cfm-dmn)# exit
RP/0/RSP0/CPU0:router(config-cfm)# exit
!
! Configure EFPs as CFM MEPs
!
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/0/0/0.0 l2transport
```

```
RP/0/RSP0/CPU0:router(config-subif)# ethernet cfm
RP/0/RSP0/CPU0:router(config-if-cfm)# mep domain GLOBAL service CustomerA mep-id 22
RP/0/RSP0/CPU0:router(config-if-cfm)# exit
RP/0/RSP0/CPU0:router(config-subif)# exit
!
! Configure the Local UNI Name
!
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0
RP/0/RSP0/CPU0:router(config-if)# ethernet uni id PE1-CustA-Slot0-Port0
RP/0/RSP0/CPU0:router(config-if)# exit
!
! Enable E-LMI on the Local UNI Physical Interface
!
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0
RP/0/RSP0/CPU0:router(config-if)# ethernet lmi
RP/0/RSP0/CPU0:router(config-if)# exit
RP/0/RSP0/CPU0:router(config)# commit
```

Where to Go Next

When you have configured an Ethernet interface, you can configure individual VLAN subinterfaces on that Ethernet interface.

For information about modifying Ethernet management interfaces for the shelf controller (SC), route processor (RP), and distributed RP, see the "Advanced Configuration and Modification of the Management Ethernet Interface on the Cisco ASR 9000 Series Router" module later in this document.

For information about IPv6 see the *Implementing Access Lists and Prefix Lists on Cisco IOS XR Software* module in the *Cisco IOS XR IP Addresses and Services Configuration Guide.*

Additional References

The following sections provide references related to implementing Gigabit and 10-Gigabit Ethernet interfaces.

Related Documents

Related Topic	Document Title
Ethernet L2VPN	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Information about user groups and task IDs	*Cisco IOS XR Interface and Hardware Component Command Reference*

Standards

Standards	Title
IEEE 802.1ag	*Connectivity Fault Management*
ITU-T Y.1731	*OAM Functions and Mechansims for Ethernet Based Networks*
MEF 16	*Metro Ethernet Forum, Technical Specification MEF 16, Ethernet Local Management Interface (E-LMI), January 2006*

MIBs

MIBs	MIBs Link
IEEE8021-CFM-MIB	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Integrated Routing and Bridging on the Cisco ASR 9000 Series Router

This module describes the configuration of Integrated Routing and Bridging (IRB) on the Cisco ASR 9000 Series Aggregation Services Routers. IRB provides the ability to exchange traffic between bridging services on the Cisco ASR 9000 Series Router and a routed interface using a Bridge-Group Virtual Interface (BVI).

Feature History for IRB

Release	Modification
Release 4.0.1	This feature was introduced on the Cisco ASR 9000 Series Router for the following line cards: • 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L) • 4-Port 10-Gigabit Ethernet Line Cards (A9K-4T-B, -E, -L) • 8-Port 10-Gigabit Ethernet DX Line Cards (A9K-8T/4-B, -E, -L) • 8-Port 10-Gigabit Ethernet Line Cards (A9K-8T-B, -E, -L) • 16-Port 10-Gigabit Ethernet Line Cards (A9K-16T/8-B, -E, -L) • 40-Port Gigabit Ethernet Line Cards (A9K-40GE-B, -E, -L)
Release 4.1.0	• Support for the following IRB environment using the Cisco ASR 9000 SIP-700 with any supported SPA as the core-facing interface was added: – Layer 3 routed traffic from the Cisco ASR 9000 SIP-700 to Layer 2 bridged interfaces on Gigabit Ethernet line cards supporting IRB. – IPv4 unicast traffic only. • Support for IPv6 unicast addressing for IRB and 6PE/6VPE support with BVI interfaces was added for the following line cards: – 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L) – 4-Port 10-Gigabit Ethernet Line Cards (A9K-4T-B, -E, -L) – 8-Port 10-Gigabit Ethernet DX Line Cards (A9K-8T/4-B, -E, -L) – 8-Port 10-Gigabit Ethernet Line Cards (A9K-8T-B, -E, -L) – 16-Port 10-Gigabit Ethernet Line Cards (A9K-16T/8-B, -E, -L) – 40-Port Gigabit Ethernet Line Cards (A9K-40GE-B, -E, -L)

Contents

Prerequisites for Configuring IRB

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring IRB, be sure that the following tasks and conditions are met:

- If you have a Cisco ASR 9000 SIP-700 installed on the core-facing side of the router, then you can support IRB for Layer 3 routed to Layer 2 bridged traffic flows for IPv4 unicast traffic, where the Layer 2 destination is one of the supported Gigabit Ethernet line cards for IRB.
- Confirm that you are configuring only the following types of Gigabit Ethernet line cards where you plan to support IRB in support of both Layer 3 to Layer 2 traffic flows and Layer 2 to Layer 3 traffic flows:
  - 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L)
  - 4-Port 10-Gigabit Ethernet Line Cards (A9K-4T-B, -E, -L)
  - 8-Port 10-Gigabit Ethernet DX Line Cards (A9K-8T/4-B, -E, -L)
  - 8-Port 10-Gigabit Ethernet Line Cards (A9K-8T-B, -E, -L)
  - 16-Port 10-Gigabit Ethernet Line Cards (A9K-16T/8-B, -E, -L)
  - 40-Port Gigabit Ethernet Line Cards (A9K-40GE-B, -E, -L)
- Know the IP addressing and other Layer 3 information to be configured on the bridge virtual interface (BVI). For more information, see the "Restrictions for Configuring IRB" section on page 187.
- Complete MAC address planning if you decide to override the common global MAC address for all BVIs.
- Be sure that the BVI network address is being advertised by running static or dynamic routing on the BVI interface.

Restrictions for Configuring IRB

Before configuring IRB, consider the following restrictions:

- Only one BVI can be configured in any bridge domain.
- The same BVI can not be configured in multiple bridge domains.

Caution If you want to support IRB on a Cisco ASR 9000 Series Router that also has a Cisco ASR 9000 SIP-700 installed, you must be sure to set up your routing configuration to prevent loss of traffic between the SIP-700 and a BVI interface. See the restrictions below for more information.

- Beginning in Cisco IOS XR Release 4.1, IRB can be implemented on supported Gigabit Ethernet line cards in a system where a Cisco ASR 9000 SIP-700 is also installed, with the following restrictions:
  - The Cisco ASR 9000 SIP-700 must be installed on the core-facing side of the router with a BVI interface configured with IPv4 addressing.
  - The Cisco ASR 9000 SIP-700 can support routing of IPv4 unicast traffic from Layer 3 to a bridged Layer 2 interface using IRB, where one of the following Gigabit Ethernet line cards is in the Layer 2 bridge domain:

    — 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L)

    — 4-Port 10-Gigabit Ethernet Line Cards (A9K-4T-B, -E, -L)

    — 8-Port 10-Gigabit Ethernet DX Line Cards (A9K-8T/4-B, -E, -L)

    — 8-Port 10-Gigabit Ethernet Line Cards (A9K-8T-B, -E, -L)

    — 16-Port 10-Gigabit Ethernet Line Cards (A9K-16T/8-B, -E, -L)

    — 40-Port Gigabit Ethernet Line Cards (A9K-40GE-B, -E, -L)

Note The reverse direction of Layer 2 bridged traffic from these line cards to Layer 3 at the Cisco ASR 9000 SIP-700 is also supported.

- The following areas are *not* supported on the BVI:
  - Access Control Lists (ACLs). However, Layer 2 ACLs can be configured on each Layer 2 port of the bridge domain.
  - IP fast reroute (FRR)
  - NetFlow
  - MoFRR
  - MPLS label switching
  - mVPNv4
  - Quality of Service (QoS)
  - Traffic mirroring
  - Unnumbered interface for BVI
  - Video monitoring (Vidmon)
- IRB with 802.1ah (BVI and Provider Backbone Bridge (PBB) should not be configured in the same bridge domain).
- PIM snooping. (Need to use selective flood.)
- VRF-aware DHCP relay is not supported.

- BVIs are supported only on bridge domains with the following characteristics:
  - The bridge domain supports single and double-tagged dot1q- and dot1ad-encapsulated EFPs with non-ambiguous or "exact match" EFP encapsulations. Single and double-tagged encapsulation can be specified as long as the **rewrite ingress tag pop symmetric** command is configured.
  - All Layer 2 tags must be removed. VLAN ranges are not supported.
  - Untagged EFPs are supported.
- The following additional functionality is *not* supported on BVI interfaces in an environment with the Cisco ASR 9000 SIP-700 at the core-facing side:
  - ARP
  - Frame Relay
  - IPv4 multicast traffic
  - IPv6 unicast and multicast traffic
  - Layer 2 traffic flows from the SIP-700 to any Layer 3 interface
  - Layer 2/Layer 3 features on BVI interfaces
  - Load intervals
  - MIBs
  - The **show adjacency details** command is not supported.

Information About Configuring IRB

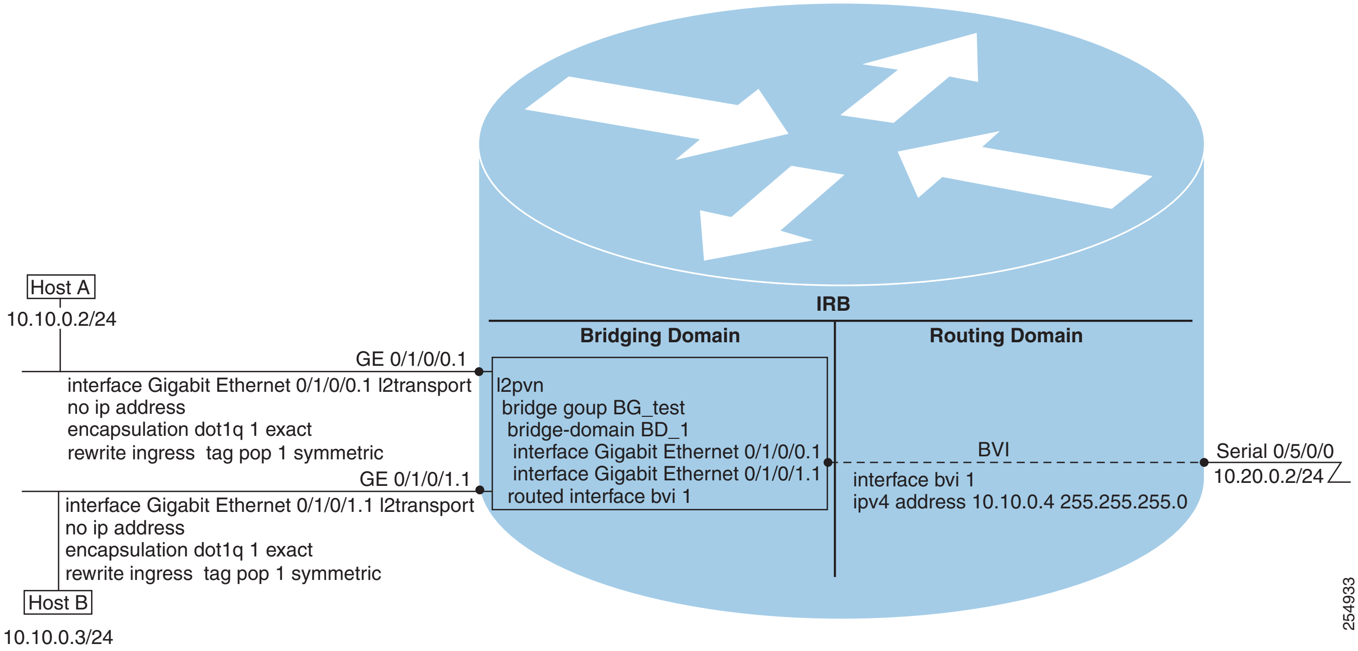
This section includes the following topics:

IRB Introduction

IRB provides the ability to route between a bridge group and a routed interface using a BVI. The BVI is a virtual interface within the router that acts like a normal routed interface. A BVI is associated with a single bridge domain and represents the link between the bridging and the routing domains on the router. To support receipt of packets from a bridged interface that are destined to a routed interface, the BVI must be configured with the appropriate IP addresses and relevant Layer 3 attributes.

In software releases before Cisco IOS XR 4.0.1 where IRB is not supported, you would need to implement a physical cabling solution to connect the egress Layer 2 bridge domain interface to a Layer 3 routing domain interface on the same Cisco ASR 9000 Series Router. In Cisco IOS XR Release 4.0.1, IRB accomplishes the same functionality using a BVI and its supporting interface and bridge group configuration shown in Figure 1.

Figure 1 IRB Functional View and Configuration Elements



```
interface Gigabit Ethernet 0/1/0/0.1 l2transport
no ip address
encapsulation dot1q 1 exact
rewrite ingress  tag pop 1 symmetric
```

```
interface Gigabit Ethernet 0/1/0/1.1 l2transport
no ip address
encapsulation dot1q 1 exact
rewrite ingress  tag pop 1 symmetric
```

```
l2pvn
 bridge goup BG_test
  bridge-domain BD_1
   interface Gigabit Ethernet 0/1/0/0.1
   interface Gigabit Ethernet 0/1/0/1.1
  routed interface bvi 1
```

```
interface bvi 1
ipv4 address 10.10.0.4 255.255.255.0
```

Bridge-Group Virtual Interface

This section includes the following information:

- BVI Introduction, page 190
- Supported Features on a BVI, page 191
- BVI MAC Address, page 191
- BVI Interface and Line Protocol States, page 191

BVI Introduction

The BVI is a virtual interface within the router that acts like a normal routed interface. The BVI does not support bridging itself, but acts as a gateway for the corresponding bridge-domain to a routed interface within the router.

Aside from supporting a configurable MAC address, a BVI supports only Layer 3 attributes, and has the following characteristics:

- Uses a MAC address taken from the local chassis MAC address pool, unless overridden at the BVI interface.
- Is configured as an interface type using the **interface bvi** command and uses an IPv4 address that is in the same subnet as the hosts on the segments of the bridged domain. The BVI also supports secondary addresses.
- The BVI identifier is independent of the bridge-domain identifier. These identifiers do not need to correlate like they do in Cisco IOS software.
- Is associated to a bridge group using the **routed interface bvi** command.

Supported Features on a BVI

- The following interface commands are supported on a BVI:
  - **arp purge-delay**
  - **arp timeout**
  - **bandwidth** (The default is 10 Gbps and is used as the cost metric for routing protocols for the BVI)
  - **ipv4**
  - **ipv6** (not supported in IRB environment with the Cisco ASR 9000 SIP-700)
  - **mac-address**
  - **mtu** (The default is 1500 bytes)
  - **shutdown**
- The BVI supports IP helper addressing and secondary IP addressing.

BVI MAC Address

By default, the Cisco ASR 9000 Series Router uses one MAC address for all BVI interfaces on the router. However, this means that the MAC address is not unique globally. If you want to override the default and specify a unique MAC address at the BVI, then you can configure it at the BVI interface.

BVI Interface and Line Protocol States

Like typical interface states on the router, a BVI has both an Interface and Line Protocol state.

- The BVI interface state is Up when the following occurs:
  - The BVI interface is created.
  - The bridge-domain that is configured with the **routed interface bvi** command has at least one available active bridge port (Attachment circuit [AC] or pseudowire [PW]).

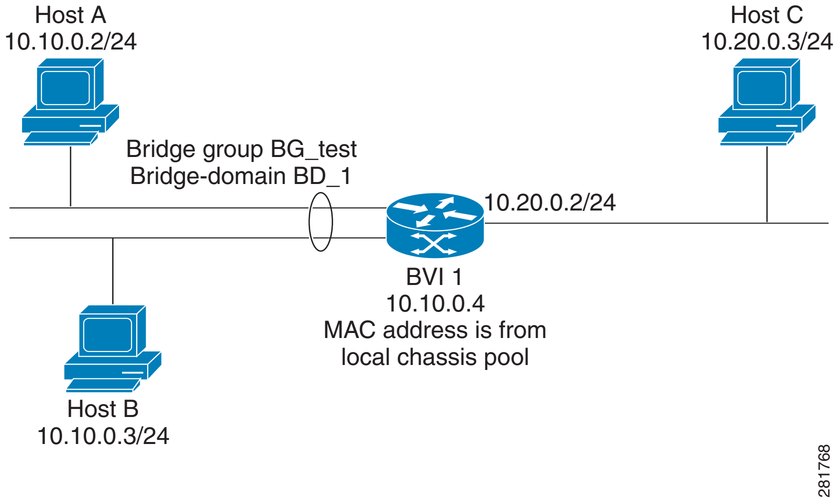
Note A BVI will be moved to the Down state if all of the bridge ports (Ethernet flow points [EFPs]) associated with the bridge domain for that BVI are down. However, the BVI will remain up if at least one pseudowire is up, even if all EFPs are down.

- The following characteristics determine when the the BVI line protocol state is up:
  - The bridge-domain is in Up state.
  - The BVI IP address is not in conflict with any other IP address on another active interface in the router.

Packet Flows Using IRB

Figure 2 shows a simplified functional diagram of an IRB implementation to describe different packet flows between Host A, B, and C. In this example, Host C is on a network with a connection to the same router. In reality, another router could be between Host C and the router shown.

Figure 2 IRB Packet Flows Between Hosts



When IRB is configured on a router, the following processing happens:

- ARP requests are resolved between the hosts and BVI that are part of the bridge domain.
- All packets from a host on a bridged interface go to the BVI if the destination MAC address matches the BVI MAC address. Otherwise, the packets are bridged.
- For packets destined for a host on a routed network, the BVI forwards the packets to the routing engine before sending them out a routed interface.
- All packets either from or destined to a host on a bridged interface go to the BVI first (unless the packet is destined for a host on the bridge domain).
- For packets that are destined for a host on a segment in the bridge domain that come in to the router on a routed interface, the BVI forwards the packet to the bridging engine, which forwards it through the appropriate bridged interface.

Packet Flows When Host A Sends to Host B on the Bridge Domain

When Host A sends data to Host B in the bridge domain on the 10.10.0.0 network, no routing occurs. The hosts are on the same subnet and the packets are bridged between their segment interfaces on the router.

Packet Flows When Host A Sends to Host C From the Bridge Domain to a Routed Interface

Using host information from Figure 2, the following occurs when Host A sends data to Host C from the IRB bridging domain to the routing domain:

- Host A sends the packet to the BVI (as long any ARP request the is resolved between the host and the BVI). The packet has the following information:
  - Source MAC address of host A.
  - Destination MAC address of the BVI.

- Since Host C is on another network and needs to be routed, the BVI forwards the packet to the routed interface with the following information:
  - IP source MAC address of Host A (10.10.0.2) is changed to the MAC address of the BVI (10.10.0.4).
  - IP destination address is the IP address of Host C (10.20.0.3).
- Interface 10.20.0.2 sees receipt of a packet from the routed BVI 10.10.0.4. The packet is then routed through interface 10.20.0.2 to Host C.

Packet Flows When Host C Sends to Host B From a Routed Interface to the Bridge Domain

Using host information from Figure 2, the following occurs when Host C sends data to Host B from the IRB routing domain to the bridging domain:

- The packet comes into the routing domain with the following information:
  - MAC source address—MAC of Host C.
  - MAC destination address—MAC of the 10.20.0.2 ingress interface.
  - IP source address—IP address of Host C (10.20.0.3).
  - IP destination address—IP address of Host B (10.10.0.3).
- When interface 10.20.0.2 receives the packet, it looks in the routing table and determines that the packet needs to be forwarded to the BVI at 10.10.0.4.
- The routing engine captures the packet that is destined for the BVI and forwards it to the BVI’s corresponding bridge domain. The packet is then bridged through the appropriate interface if the destination MAC address for Host B appears in the bridging table, or is flooded on all interfaces in the bridge group if the address is not in the bridging table.

Supported Environments for IRB

The following environments and configuration elements are supported with IRB on the Cisco ASR 9000 Series Router:

- Configuration of one BVI per bridge domain.
- Virtual Private LAN Service (VPLS) virtual forwarding instance (VFI) configuration associated with a bridge domain configured with a BVI.
- BGP PIC edge for BVI-based prefixes.
- Traffic forwarding for the BVI using Open Shortest Path First (OSPF), Intermediate System-to-Intermediate System (IS-IS), Routing Information Protocol Version 2 (RIPv2), and Border Gateway Protocol (BGP).
- Internet Group Management Protocol (IGMP) static groups.
- Dynamic Host Configuration Protocol (DHCP) relay agent. When DHCP relay is used from an aggregation node to obtain an IP address, the default gateway will be the IP address configured on the BVI. The BVI IP address should be in a common subnet as the DHCP pool that is being used by the aggregation node to assign IP addresses.
- Virtual Router Redundancy Protocol (VRRP) configuration and priority.
- Hot Standby Router Protocol (HSRP).
- Up to 255 VRRF/HSRP VMAC per BVI interface.

- Bridging of non-IP packets on a bridge domain configured with a BVI.
- Parity with stateful protocol support as currently supported on Layer 3 subinterfaces on the Cisco ASR 9000 Series Router.
- IP SLA support as currently supported on Layer 3 subinterfaces on the Cisco ASR 9000 Series Router.
- Load balancing of BVIs as ECMP paths (up to 32 paths).
- Interface-MIB.
- Packet counters for BVI interfaces.
- Multi-chassis link aggregation (LAG) on link bundles that are members of a bridge domain that uses a BVI.

The following sections document additional IPv4- and IPv6-specific environments supported for IRB:

- Additional IPv4-Specific Environments Supported for IRB, page 194
- Additional IPv6-Specific Environments Supported for IRB, page 194

Additional IPv4-Specific Environments Supported for IRB

- Configuration of up to a maximum of 2000 BVIs.
- Up to a maximum of128k IPv4 adjacencies.
- Layer 3 IP multicast, with ability to take ingress IP multicast traffic and bridge it to multiple Layer 2 subinterfaces (Ethernet flow points) on a bridge domain that are part of multicast groups.

Note Not supported when used with the Cisco ASR 9000 SIP-700 at core-facing side.

- VRFs for IPv4 (Per-VPN label VRFs only—not per prefix).

Additional IPv6-Specific Environments Supported for IRB

- Configuration of up to a maximum of 2000 BVIs, with up to 512 of these BVIs that can support IPv6 addressing.
- Up to a maximum of 5k IPv6 adjacencies.
- Cisco IPv6 Provider Edge Router over MPLS (6PE) and IPv6 VPN Provider Edge (6VPE) support with BVI interfaces at the customer edge (CE)-facing side of the Cisco ASR 9000 Series Router as the PE device with the following restrictions:
  - Supported by the following line cards on the PE devices:

    — 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L)

    — 4-Port 10-Gigabit Ethernet Line Cards (A9K-4T-B, -E, -L)

    — 8-Port 10-Gigabit Ethernet DX Line Cards (A9K-8T/4-B, -E, -L)

    — 8-Port 10-Gigabit Ethernet Line Cards (A9K-8T-B, -E, -L)

    — 16-Port 10-Gigabit Ethernet Line Cards (A9K-16T/8-B, -E, -L)

    — 40-Port Gigabit Ethernet Line Cards (A9K-40GE-B, -E, -L)
  - Up to 512 BVIs with IPv6 addressing can be supported.

- Only per-VRF label allocation is supported (using the **label-allocation-mode per-vrf** command).

For a configuration example, see the "6PE/6VPE With BVI Configuration: Example" section on page 206.

How to Configure IRB

This section includes the following configuration tasks:

- Configuring the Bridge Group Virtual Interface, page 195 (Required)
- Configuring the Layer 2 AC Interfaces, page 197 (Required)
- Configuring a Bridge Group and Assigning Interfaces to a Bridge Domain, page 199 (Required)
- Associating the BVI as the Routed Interface on a Bridge Domain, page 201 (Required)
- Displaying Information About a BVI, page 203 (Optional)

Configuring the Bridge Group Virtual Interface

To configure a BVI, complete the following steps.

Configuration Guidelines

Consider the following guidelines when configuring the BVI:

- The BVI must be assigned an IPv4 or IPv6 address that is in the same subnet as the hosts in the bridged segments.
- If the bridged network has multiple IP networks, then the BVI must be assigned secondary IP addresses for each network.

SUMMARY STEPS

1. **configure**
2. **interface bvi** *identifier*
3. **ipv4 address** *ipv4-address mask* [**secondary**]

   or

   ipv6 address *ipv6-prefix***/***prefix-length* [**eui-64**] [**route-tag** *route-tag value*]
4. **arp purge-delay** *seconds*
5. **arp timeout** *seconds*
6. **bandwidth** *rate*
7. **mac-address** *value1***.***value2***.***value3*
8. **mtu** *bytes*
9. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface bvi** *identifier* **Example:** `RP/0/RSP0/CPU0:router(config)# interface bvi 1`	Specifies or creates a BVI, where *identifier* is a number from 1 to 65535.
Step 3	**ipv4 address** *ipv4-address mask* [**secondary**] **ipv6 address** *ipv6-prefix***/***prefix-length* [**eui-64**] [**route-tag** *route-tag value*] **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.10.0.4 255.255.255.0`	Specifies a primary or secondary IPv4 address or an IPv6 address for an interface.
Step 4	**arp purge-delay** *seconds* **Example:** `RP/0/RSP0/CPU0:router(config-if)#arp purge-delay 120`	(Optional) Specifies the amount of time (in *seconds*) to delay purging of Address Resolution Protocol (ARP) table entries when the interface goes down. The range is 1 to 65535. The default is no purge delay is configured.
Step 5	**arp timeout** *seconds* **Example:** `RP/0/RSP0/CPU0:router(config-if)# arp timeout 12200`	(Optional) Specifies how long dynamic entries learned on the interface remain in the ARP cache. The range is 30 to 2144448000 seconds. The default is 14,400 seconds (4 hours).
Step 6	**bandwidth** *rate* **Example:** `RP/0/RSP0/CPU0:router(config-if)# bandwidth 1000000`	(Optional) Specifies the amount of bandwidth (in kilobits per second) to be allocated on the interface. This number is used as the cost metric in routing protocols for the BVI. The range is 0 to 4294967295. The default is 10000000 (10 Gbps).
Step 7	**mac-address** *value1***.***value2***.***value3* **Example:** `RP/0/RSP0/CPU0:router(config-if)# mac-address 1111.2222.3333`	(Optional) Specifies the 48-bit MAC address for the BVI as three dotted-hexadecimal values, and overrides use of the default MAC address. The range for each value is 0000 to ffff. A MAC address of all 0s is not supported.

	Command or Action	Purpose
Step 8	**mtu** *bytes* **Example:** `RP/0/RSP0/CPU0:router(config-if)# mtu 2000`	(Optional) Specifies the maximum transmission unit (MTU) size for packets on the interface. The range is 64 to 65535. The default is 1514.

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Layer 2 AC Interfaces

To configure the Layer 2 AC interfaces for routing by a BVI, complete the following steps.

Prerequisites

The interfaces to be configured as Layer 2 ACs in the bridge domain and routed by a BVI must be located on the following types of cards supporting IRB on the Cisco ASR 9000 Series Router:

- 2-Port 10-Gigabit Ethernet, 20-Port Gigabit Ethernet Combination Line Cards (A9K-2T20GE-B and A9K-2T20GE-L)
- 4-Port 10-Gigabit Ethernet Line Cards (A9K-4T-B, -E, -L)
- 8-Port 10-Gigabit Ethernet DX Line Cards (A9K-8T/4-B, -E, -L)
- 8-Port 10-Gigabit Ethernet Line Cards (A9K-8T-B, -E, -L)
- 40-Port Gigabit Ethernet Line Cards (A9K-40GE-B, -E, -L)

SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id*[.*subinterface*] **l2transport**
3. **no ip address**

4. **encapsulation dot1q** *vlan-id* **exact**

   or

   encapsulation dot1ad *vlan-id* **dot1q** *vlan-id*

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

6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*[.*subinterface*] **l2transport** **Example:** `RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/1/0/0.1 l2transport`	Enables Layer 2 transport mode on a Gigabit Ethernet or 10-Gigabit Ethernet interface or subinterface and enters interface or subinterface configuration mode, where *interface-path-id* is specified as the *rack***/***slot***/***module***/***port* location of the interface and **.***subinterface* is the optional subinterface number.
Step 3	**encapsulation dot1q** *vlan-id* [**exact**] or **encapsulation dot1ad** *vlan-id* **dot1q** *vlan-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1q 1 exact`	(Optional) Specifies IEEE 802.1q encapsulation on the specified VLAN only.

	Command or Action	Purpose
Step 4	**rewrite ingress tag pop** {**1** | **2**} **symmetric** **Example:** `RP/0/RSP0/CPU0:router(config-if)# rewrite ingress tag pop 1 symmetric`	(Required if VLAN tagging configured) Specifies that one or two tags (depending on the network configuration) should be removed from frames arriving at the ingress interface to the bridge domain. **Note** If configuring double tags using dot1ad and dot1q encapsulation, you need to use the **rewrite ingress tag pop 2 symmetric** command.

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring a Bridge Group and Assigning Interfaces to a Bridge Domain

To configure a bridge group and assign interfaces to a bridge domain, complete the following steps.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *name*
4. **bridge-domain** *name*
5. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id*[**.***subinterface*]
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**l2vpn** **Example:** `RP/0/RSP0/CPU0:router(config)# l2vpn`	Enters L2VPN configuration mode.
Step 3	**bridge group** *bridge-group-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 10`	Creates a bridge group and enters L2VPN bridge group configuration mode.
Step 4	**bridge-domain** *bridge-domain-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain BD_1`	Creates a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

	Command or Action	Purpose
Step 5	**interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*[.*subinterface*] **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/1/0/0.1`	Associates the Gigabit Ethernet and 10-Gigabit Ethernet interface with the specified bridge domain and enters L2VPN bridge group bridge domain attachment circuit configuration mode, where *interface-path-id* is specified as the *rack/slot/module/port* location of the interface and *.subinterface* is the optional subinterface number. Repeat this step for as many interfaces as you want to associate with the bridge domain.

Step 6	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Associating the BVI as the Routed Interface on a Bridge Domain

To associate the BVI as the routed interface on a bridge domain, complete the following steps.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge-group-name*
4. **bridge-domain** *bridge-domain-name*
5. **routed interface bvi** *identifier*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# l2vpn`
		Enters L2VPN configuration mode.

Step 3	**`bridge group`** *`bridge-group-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group BG_test`
		Creates a bridge group and enters L2VPN bridge group configuration mode.

Step 4	**`bridge-domain`** *`bridge-domain-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 1`
		Creates a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

Step 5	**`routed interface bvi`** *`identifier`*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# routed interface bvi 1`
		Associates the specified BVI as the routed interface for the interfaces assigned to the bridge domain.

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Displaying Information About a BVI

To display information about BVI status and packet counters, use the following commands:

Command	Purpose
show interfaces bvi *identifier* [**accounting** | **brief** | **description** | **detail**]	Displays interface status, line protocol state, and packet counters for the specified BVI.
show adjacency bvi *identifier* [**detail** | **remote**]	Displays packet and byte transmit counters per adjacency to the specified BVI.
show l2vpn bridge-domain detail	Displays the reason that a BVI is down.

Configuration Examples for IRB

This section provides the following configuration examples:

- Basic IRB Configuration: Example, page 203
- IRB Using ACs With VLANs: Example, page 204
- IPv4 Addressing on a BVI Supporting Multiple IP Networks: Example, page 204
- Comprehensive IRB Configuration with BVI Bundle Interfaces and Multicast Configuration: Example, page 205
- IRB With BVI and VRRP Configuration: Example, page 206

Basic IRB Configuration: Example

The following example shows how to perform the most basic IRB configuration:

```
! Configure the BVI and its IPv4 address
!
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface bvi 1
RP/0/RSP0/CPU0:router(config-if))# ipv4 address 10.10.0.4 255.255.255.0
RP/0/RSP0/CPU0:router(config-if))# exit
!
! Configure the Layer 2 AC interface
!
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/1/0/0 l2transport
RP/0/RSP0/CPU0:router(config-if))# exit
!
! Configure the L2VPN bridge group and bridge domain and assign interfaces
!
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 10
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/1/0/0
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-if)# exit
!
! Associate a BVI to the bridge domain
!
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# routed interface bvi 1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit
```

IRB Using ACs With VLANs: Example

The following example shows how to configure IRB on a bridge domain with Layer 2 ACs using 802.1q-encapsulated VLANs:

```
! Configure the BVI and its IPv4 address
!
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface bvi 1
RP/0/RSP0/CPU0:router(config-if))# ipv4 address 10.10.0.4 255.255.255.0
RP/0/RSP0/CPU0:router(config-if))# exit
!
! Configure the Layer 2 AC interfaces using dot1q encapsulation on a VLAN
!
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/1/0/0.1 l2transport
RP/0/RSP0/CPU0:router(config-if))# no ip address
RP/0/RSP0/CPU0:router(config-if))# encapsulation dot1q 1 exact
RP/0/RSP0/CPU0:router(config-if))# rewrite ingress tag pop 1 symmetric
RP/0/RSP0/CPU0:router(config-if))# exit
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/1/0/1.1 l2transport
RP/0/RSP0/CPU0:router(config-if))# no ip address
RP/0/RSP0/CPU0:router(config-if))# encapsulation dot1q 1 exact
RP/0/RSP0/CPU0:router(config-if))# rewrite ingress tag pop 1 symmetric
RP/0/RSP0/CPU0:router(config-if))# exit
!
! Configure the L2VPN bridge group and bridge domain and assign interfaces
!
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 10
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/1/0/0.1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/1/0/1.1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-if)# exit
!
! Associate a BVI to the bridge domain
!
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# routed interface bvi 1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit
```

IPv4 Addressing on a BVI Supporting Multiple IP Networks: Example

The following example shows how to configure secondary IPv4 addresses on a BVI that supports bridge domains for the 10.10.10.0/24, 10.20.20.0/24, and 10.30.30.0/24 networks. In this example, the BVI must have an address on each of the bridge domain networks:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface bvi 1
RP/0/RSP0/CPU0:router(config-if))# ipv4 address 10.10.10.4 255.255.255.0
RP/0/RSP0/CPU0:router(config-if))# ipv4 address 10.20.20.4 255.255.255.0 secondary
RP/0/RSP0/CPU0:router(config-if))# ipv4 address 10.30.30.4 255.255.255.0 secondary
RP/0/RSP0/CPU0:router(config-if))# commit
```

Comprehensive IRB Configuration with BVI Bundle Interfaces and Multicast Configuration: Example

The following example shows a more comprehensive router configuration with IRB and BVI multicast support:

```
interface Bundle-Ether25
 ipv4 address 10.21.0.2 255.255.255.0
!
interface Loopback0
 ipv4 address 10.5.5.5 255.255.255.255
!
interface GigabitEthernet0/0/0/1
 negotiation auto
!
interface GigabitEthernet0/0/0/1.1 l2transport
 encapsulation dot1q 1
 rewrite ingress tag pop 1 symmetric
!
interface GigabitEthernet0/0/0/1.2 l2transport
 encapsulation dot1q 2
 rewrite ingress tag pop 1 symmetric
!

interface GigabitEthernet0/0/0/9
 bundle id 25 mode active
!
interface GigabitEthernet0/0/0/19
 bundle id 25 mode active
!
interface GigabitEthernet0/0/0/29
 bundle id 25 mode active
!

interface GigabitEthernet0/0/0/39
 bundle id 25 mode active

interface BVI1
 ipv4 address 10.1.1.1 255.255.255.0
!
interface BVI2
 ipv4 address 10.1.2.1 255.255.255.0

router ospf 100
 router-id 10.5.5.5
 area 0
  interface Bundle-Ether25
    interface Loopback0
    interface BVI1
   interface BVI2
  !
l2vpn
 bridge group IRB
  bridge-domain IRB1
   igmp snooping profile IRB_SNOOP
   interface GigabitEthernet0/0/0/1.1
   !
   routed interface BVI1
  !
  bridge-domain IRB2
   igmp snooping profile IRB_SNOOP
   interface GigabitEthernet0/0/0/1.2
   !
   routed interface BVI2

multicast-routing
 address-family ipv4
  interface all enable
igmp snooping profile IRB_SNOOP
 report-suppression disable
!
router pim
 address-family ipv4
  rp-address 10.10.10.10
```

IRB With BVI and VRRP Configuration: Example

The following example shows a partial router configuration for the relevant configuration areas for IRB support of a BVI and VRRP:

Note VRRPv6 is also supported.

```
l2vpn
 bridge group IRB
  bridge-domain IRB-EDGE
   interface GigabitEthernet0/0/0/8
!
   routed interface BVI 100
!
interface GigabitEthernet0/0/0/8
  l2transport
!
interface BVI 100
 ipv4 address 10.21.1.1 255.255.255.0
!
router vrrp
 interface BVI 100
  vrrp 1 ipv4 10.21.1.100
  vrrp 1 priority 100
 !
```

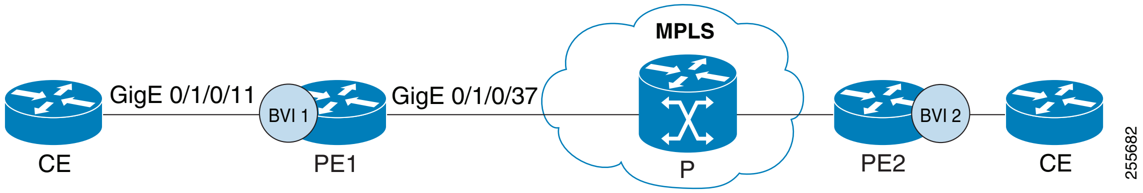
6PE/6VPE With BVI Configuration: Example

The following example shows how to configure an MPLS 6PE/6VPE environment using BVIs at the CE-facing sides of the Cisco ASR 9000 Series Router as the PE devices. For more information about Cisco 6PE/6VPE and its configuration, see the "Implementing IPv6 VPN Provider Edge Transport Over MPLS" chapter of the *Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide*.

Note This environment is only supported using IRB with the supported Gigabit Ethernet line cards on the Cisco ASR 9000 Series Router. It is not supported with the Cisco ASR 9000 SIP-700 SPAs.

Figure 3 shows the location of the BVI interfaces (green icons) on the Cisco ASR 9000 Series Routers as the PE1 and PE2 devices.

Figure 3 BVI Interfaces on the CE-Facing Sides in an MPLS 6PE/6VPE Network



The following example is a sample configuration only for the Cisco ASR 9000 Series Router (PE1) device with a BVI interface numbered 1 on the CE-facing side, and a non-BVI interface (Gigabit Ethernet 0/1/0/37) on the core-facing side. A similar configuration would apply to the PE2 device:

```
! Be sure to configure IPv6 unicast address families
!
vrf 1
address-family ipv6 unicast
  import route-target
   100:2
  export route-target
   100:2

interface Loopback0
 ipv4 address 10.11.11.11/32
!
! Configure the BVI interface to participate in the VRF
! and with an IPv6 address.
!
interface BVI1
 vrf 1
 ipv6 address 2001:DB8:1/32
!
! Assign the Gigabit Ethernet CE-facing interface to the
! L2VPn bridge domain where the routed BVI interface is also associated.
!
l2vpn
 bridge group 1
  bridge-domain 1
   interface Gigabit Ethernet 0/1/0/11
 routed interface BVI1
!
! Configure OSPF routing for the BVI interface for
! advertisement of its IPv6 address.
!
router ospfv3 1
 graceful-restart
 redistribute bgp 1
 area 1
  interface BVI1
  interface Loopback0
!
! Configure BGP routing and be sure to specify the
! IPv6 unicast address family.
! Note that the per-VRF label allocation mode is required
! and is the only supported label allocation mode.
!
router bgp 1
```

```
 bgp router-id 10.11.11.11
 bgp redistribute-internal
 bgp graceful-restart

 address-family ipv6 unicast
  redistribute ospfv3 1 match internal external
  label-allocation-mode per-vrf
  allocate-label all
 !
 address-family vpnv6 unicast
 !
 neighbor 10.11.12.12
  remote-as 1
  update-source Loopback0
  address-family ipv6 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
  address-family ipv6 labeled-unicast
  !
  address-family vpnv6 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 vrf 1
  rd 100:2
 label-allocation-mode per-vrf
  address-family ipv6 unicast
   redistribute connected

mpls ldp
 router-id 10.11.11.11
 graceful-restart
 interface Gigabit Ethernet 0/1/0/37
```

Additional References

The following sections provide references related to configuring IRB on the Cisco ASR 9000 Series Router.

Related Documents

Related Topic	Document Title
Ethernet L2VPN	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*
Cisco IOS XR master command reference	*Cisco ASR 9000 Series Aggregation Services Router Master Command Listing, Release 4.0*
Cisco IOS XR interface configuration commands	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*

Related Topic	Document Title
Cisco IOS XR multicast configuration	*Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide*
MPLS Layer 3 VPN configuration	*Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
IF-MIB	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Link Bundling on the Cisco ASR 9000 Series Router

This module describes the configuration of link bundle interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

A link bundle is a group of one or more ports that are aggregated together and treated as a single link.

Each bundle has a single MAC, a single IP address, and a single configuration set (such as ACLs). POS link bundles do not have mac address, only ethernet link bundles have mac address.

Note The Cisco ASR 9000 Series Router supports both Layer 2 and Layer 3 Link Bundles. If the Link Bundle is a Layer 3 interface, an IP address is required. If the Link Bundle is a Layer 2 interface, an IP address is not required. A Link Bundle on the Cisco ASR 9000 Series Router may contain Layer 2 and Layer 3 subinterfaces within it. In which case, the Layer 3 subinterfaces require IP addresses, but the Link Bundle interface does not require an IP address. POS Link bundling is supported only on Layer 3 link bundles.

The Cisco ASR 9000 Series Router supports bundling for the following types of interfaces:

- Ethernet interfaces and
- POS interfaces on the ASR 9000 SIP-700 line card.

Feature History for Configuring Link Bundling

Release	Modification
Release 3.7.2	This feature was introduced on the Cisco ASR 9000 Series Router.
Release 3.9.0	Support for load balancing was added. Bundle member links are put into new err-disable link interface status and admin-down protocol state when a bundle interface is shut down.
Release 3.9.1	Support for Layer 3 load balancing on Layer 2 link bundles was added.
Release 4.0.0	The following support was added: • Up to a maximum of 64 member links per bundle. • IPv6 addressing. • Multichassis Link Aggregation.
Release 4.0.1	Support for Dynamic Load Balancing for Link Aggregation (LAG) members was added. The **hw-module load-balance bundle l2-service l3-params** command is replaced by the **load-balancing flow** command in L2VPN configuration mode. For more information see the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* and *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*.
Release 4.1.0	Support for Multi-Gigabit Service Control Point was added.
Release 4.2.0	Support for Link bundling for POS interfaces was added.

Contents

This module includes the following sections:

Prerequisites for Configuring Link Bundling

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

The prerequisites for link bundling depend on the platform on which you are configuring this feature. This section includes the following information:

Prerequisites for Configuring Link Bundling on Cisco ASR 9000 Series Router

Before configuring Link Bundling, be sure that the following tasks and conditions are met:

- You know the interface IP address (Layer 3 only).
- You know which links should be included in the bundle you are configuring.
- If you are configuring an Ethernet link bundle, you have at least one of the following Ethernet line cards installed in the router:
  - 4-port 10-Gigabit Ethernet line card
  - 8-port 10-Gigabit Ethernet line card
  - 40-port Gigabit Ethernet line card
  - 24-Port 10GE DX Line Card, Packet Transport Optimized with SFP+ optics
  - 24-Port 10GE DX Line Card, Service Edge Optimized with SFP+ optics
  - 20-Port GE Modular Port Adapter (MPA) with SFP optics
  - 4-Port 10GE Modular Port Adapter (MPA) with SFP+ optics
  - 2-Port 100GE DX Line Card, Packet Transport Optimized with CFP optics
  - 2-Port 100GE DX Line Card, Service Edge Optimized with CFP optics
- If you are configuring a POS link bundle, you must have this line card installed in the router:
  - ASR 9K-SIP-700 line card
- The POS link bundling feature is supported on the following shared port adaptors(SPA):
  - 2-port OC-48 POS/SDH SPA
  - 4-port OC-48 POS/SDH SPA
  - 1-port OC-192 POS/XFP SPA
  - 4-port OC-3 POS-V2 SPA
  - 8-port OC-3 POS/SDH SPA
  - 8-port OC-12 POS/SDH SPA

Note For more information about physical interfaces, PLIMs, and modular services cards, refer to the *Cisco ASR 9000 Series Router Hardware Installation Guide*.

Information About Configuring Link Bundling

To configure link bundling, you must understand the following concepts:

Link Bundling Overview

The Link Bundling feature allows you to group multiple point-to-point links together into one logical link and provide higher bidirectional bandwidth, redundancy, and load balancing between two routers. A virtual interface is assigned to the bundled link. The component links can be dynamically added and deleted from the virtual interface.

The virtual interface is treated as a single interface on which one can configure an IP address and other software features used by the link bundle. Packets sent to the link bundle are forwarded to one of the links in the bundle.

A link bundle is simply a group of ports that are bundled together and act as a single link. The advantages of link bundles are as follows:

- Multiple links can span several line cards to form a single interface. Thus, the failure of a single link does not cause a loss of connectivity.
- Bundled interfaces increase bandwidth availability, because traffic is forwarded over all available members of the bundle. Therefore, traffic can flow on the available links if one of the links within a bundle fails. Bandwidth can be added without interrupting packet flow.

All the individual links within a single bundle must be of the same type and the same speed.

For example, a bundle can contain all Ethernet interfaces, or it can contain all POS interfaces, but it cannot contain Ethernet and POS interfaces at the same time.

Cisco IOS XR software supports the following methods of forming bundles of Ethernet interfaces:

- IEEE 802.3ad—Standard technology that employs a Link Aggregation Control Protocol (LACP) to ensure that all the member links in a bundle are compatible. Links that are incompatible or have failed are automatically removed from a bundle.
- EtherChannel or POS Channel—Cisco proprietary technology that allows the user to configure links to join a bundle, but has no mechanisms to check whether the links in a bundle are compatible.(EtherChannel applies to Ethernet interfaces, and POS Channel applies to POS interfaces.)

Features and Compatible Characteristics of Ethernet Link Bundles

The following list describes the properties and limitations of ethernet link bundles on Cisco ASR 9000 Series Routers:

- Any type of Ethernet interfaces can be bundled, with or without the use of LACP (Link Aggregation Control Protocol).

- Bundle membership can span across several line cards that are installed in a single router.An ethernet link bundle can support a maximum of 64 physical links. If you add more than 64 links to a bundle, only 64 of the links are in distributing state, and the remaining links are in waiting state.
- A single Cisco ASR 9000 Series Router supports a maximum of 256 bundles.
- All the individual links within a single ethernet link bundle must be the same speed.
- Physical layer and link layer configuration are performed on individual member links of a bundle.
- Configuration of network layer protocols and higher layer applications is performed on the bundle itself.
- IPv4 and IPv6 addressing is supported on ethernet link bundles.
- A bundle can be administratively enabled or disabled. Beginning in Cisco IOS XR Release 3.9.0, when you shut down a bundle interface, the member links are put into err-disable link interface status and admin-down line protocol state. You can show the status of a bundle interface and its members using the **show interfaces** command.
- Each individual link within a bundle can be administratively enabled or disabled.
- Ethernet link bundles are created in the same way as Ethernet channels, where the user enters the same configuration on both end systems.
- Each individual member link within a bundle has unique MAC address.
- When LACP configured, each link within a bundle can be configured to allow different keepalive periods on different members..
- Load balancing (the distribution of data between member links) is done by flow instead of by packet. Data is distributed to a link in proportion to the bandwidth of the link in relation to its bundle.
- QoS is supported and is applied proportionally on each bundle member.
- Link layer protocols, such as CDP and HDLC keepalives, work independently on each link within a bundle.
- Upper layer protocols, such as routing updates and hellos, are sent over any member link of an ethernet interface bundle.
- All links within a single bundle must terminate on the same two systems. Both systems must be directly connected.
- Bundled interfaces are point-to-point.
- A link must be in the up state before it can be in distributing state in a bundle.
- All links within a single bundle must be configured either to run 802.3ad (LACP) or Etherchannel (non-LACP). Mixed links within a single bundle are not supported.
- A bundle interface can contain physical links and VLAN subinterfaces only. Tunnels cannot be bundle members.
- Access Control List (ACL) configuration on link bundles is identical to ACL configuration on regular interfaces.
- Multicast traffic is load balanced over the members of a bundle. For a given flow, internal processes select the member link and all traffic for that flow is sent over that member.

Characteristics of POS Link Bundles in Cisco ASR 9000 Series Router

This section lists the properties of POS link bundles that are specific to Cisco ASR 9000 Series Router:

- Each bundle has to be configured between a pair of directly connected systems.

- All members of a bundle must be POS.
- The Cisco ASR 9000 SIP-700 line card can physically accomodate upto 32 POS link bundles.
- POS link bundling can support up to 32 physical links if they are in the same speed. If links are in different speed, it cannot reach 32 physical links.
- Only physical interfaces can become bundle members.
- All bundles must be statically configured.
- Only cHDLC encapsulation type is currently supported on POS Link Bundle.
- Only POS SPA is supported for POS Link Bundling and not channelized SPA.
- Upper layer protocols, such as routing updates and hellos, are sent over through the bundle interface.
- Bandwidths for policers and queues must be in percentage and not in absolute values.
- Queue-limit must be in time unit and not in bytes.
- For POS link bundles, different link speeds are allowed within a single bundle, with a maximum of four times the speed difference between the members of the bundle. This means that only up to 4 times the bandwidth ratio is supported.

Restrictions of POS Link Bundles in Cisco ASR 9000 Series Router

This section lists the limitations of POS link bundles that are specific to Cisco ASR 9000 Series Router:

- LACP is not supported for POS link bundles in Cisco IOS XR Release 4.2.0.
- IPv6 and ACL are not supported for POS link bundels in Cisco IOS XR Release 4.2.0.
- Multicast routing is not supported for POS link bundles in Cisco IOS XR Release 4.2.0.

Link Aggregation Through LACP

The optional Link Aggregation Control Protocol (LACP) is defined in the IEEE 802 standard. LACP communicates between two directly connected systems (or peers) to verify the compatibility of bundle members. For the Cisco ASR 9000 Series Router, the peer can be either another router or a switch. LACP monitors the operational state of link bundles to ensure the following:

- All links terminate on the same two systems.
- Both systems consider the links to be part of the same bundle.
- All links have the appropriate settings on the peer.

LACP transmits frames containing the local port state and the local view of the partner system’s state. These frames are analyzed to ensure both systems are in agreement.

IEEE 802.3ad Standard

The IEEE 802.3ad standard typically defines a method of forming Ethernet link bundles.

For each link configured as bundle member, the following information is exchanged between the systems that host each end of the link bundle:

- A globally unique local system identifier
- An identifier (operational key) for the bundle of which the link is a member

- An identifier (port ID) for the link
- The current aggregation status of the link

This information is used to form the link aggregation group identifier (LAG ID). Links that share a common LAG ID can be aggregated. Individual links have unique LAG IDs.

The system identifier distinguishes one router from another, and its uniqueness is guaranteed through the use of a MAC address from the system. The bundle and link identifiers have significance only to the router assigning them, which must guarantee that no two links have the same identifier, and that no two bundles have the same identifier.

The information from the peer system is combined with the information from the local system to determine the compatibility of the links configured to be members of a bundle.

Bundle MAC addresses in the Cisco ASR 9000 Series Router come from a set of reserved MAC addresses in the backplane.This MAC address stays with the bundle as long as the bundle interface exists. The bundle uses this MAC address until the user configures a different MAC address. The bundle MAC address is used by all member links when passing bundle traffic. Any unicast or multicast addresses set on the bundle are also set on all the member links.

Note We recommend that you avoid modifying the MAC address, because changes in the MAC address can affect packet forwarding.

Multichassis Link Aggregation

The Multichassis Link Aggregation (MC-LAG) feature provides an end to end interchassis redundancy solution for the Carrier Ethernet Networks. MC-LAG involves two devices collaborating to act as a single LAG from the perspective of a (third) connected device, thus providing device-level as well as link-level redundancy.

To achieve this, two devices co-ordinate with each other to present a single LACP bundle (spanning the two devices) to a partner device. Only one of the devices forwards traffic at any one time, eliminating the risk of forwarding loops. When a failure occurs, these devices coordinate to perform a switchover, changing the device on which traffic is being forwarded by manipulating the link LACP states.

The existing pseudowire redundancy in the core network coordinates with the redundancy in the access network based on:

- Multichassis Link Aggregation Control Protocol (mLACP)
- Interchassis Communication Protocol (ICCP)

The mLACP protocol defines the expected behavior between the two devices and uses the Interchassis Control Protocol (ICCP) to exchange TLVs and identify peer devices to operate with. At the edge of a provider's network, a simple customer edge (CE) device that only supports standard LACP is connected to two provider edge (PE) devices. Thus the CE device is dual-homed, providing better L2 redundancy from the provider's side. In mLACP terminology, the CE device is referred to as a dual-homed device (DHD) and each PE device is known as a point of attachment (POA). The POA forwarding traffic for the bundle is the active device for that bundle, while the other POA is the standby device.

Failure Cases

MC-LAG provides redundancy, switching traffic to the unaffected POA while presenting an unchanged bundle interface to the DHD, for the following failure events:

- Link failure: A port or link between the DHD and one of the POAs fails.
- Device failure: Meltdown or reload of one of the POAs, with total loss of connectivity (to the DHD, the core and the other POA).
- Core isolation: A POA loses its connectivity to the core network, and therefore is of no value, being unable to forward traffic to or from the DHD.

A loss of connectivity between the POAs leads both devices to assume that the other has experienced device failure, causing them to attempt to take on the Active role. This is known as a split brain scenario and can happen in either of the following cases:

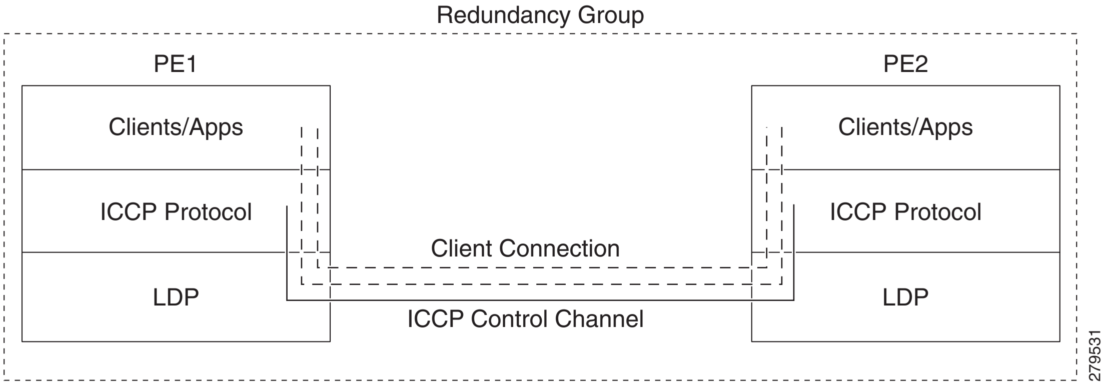
- All other connectivity remains; only the link between POAs is lost.
- One POA is isolated from the core network (i.e. a core isolation scenario where the connection between the two POAs was over the core network).

MC-LAG by itself does not provide a means to avoid this situation; resiliency in the connection between the POAs is a requirement. The DHD is given the responsibility of mitigating the problem by setting a limit on the number of links, within the bundle, that can be active. As such only the links connected to one of the POAs can be active at any one point of time.

Interchassis Communication Protocol

Figure 4 shows the graphical representation of the Interchassis Communication Protocol (ICCP).

Figure 4 ICCP Protocol



Two POAs communicate with each other over an LDP link using the Interchassis Communication Protocol (ICCP). ICCP is an LDP based protocol wherein an LDP session is created between the POAs in a redundancy group, and the ICCP messages are carried over that LDP session. The PE routers in a redundancy group may be a single-hop (directly connected) or a multi-hop away from one another. The ICCP protocol manages the setup and controls the redundancy groups. It also establishes, maintains, and tears down ICCP connections. The ICCP protocol uses route-watch to monitor the connectivity to the PEs in a given redundancy group. It is also responsible for tracking core isolation failures. It notifies all client applications of failure (core isolation and active PE failure).

To operate ICCP, the devices are configured as members of redundancy groups (RGs).

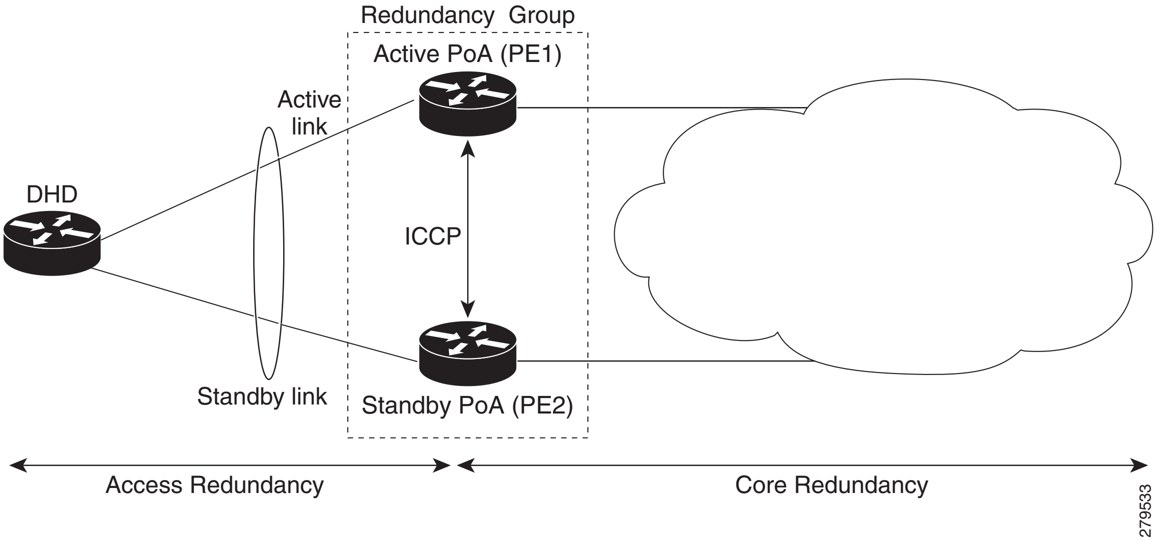
Note In the mLACP configuration, two devices are configured to be members of each RG (until a device-level failure occurs leaving only a single member). However, each device can be a member of more than one RG.

In each redundancy group, a POA's mLACP peer is the other POA in that group, with which it communicates using mLACP over ICCP. For each bundle, the POA and DHD at each end are LACP partners, communicating using the standard LACP protocol.

Access Network Redundancy Model

The Multichassis Link Aggregation Control Protocol (mLACP) based redundancy between the customer edge device (CE) or access network and the provider edge (PE) device is achieved by allowing the CE to be connected to two PE routers. The two PE routers synchronize the data through ICCP; therefore they appear as a single device to the CE.

Figure 5 mLACP/ICCP Redundancy Model



The CE is also called dual-homed device (DHD) and the PE is also called point of attachment (POA). The pair of POAs that is connected to the single DHD forms a redundancy group (RG).

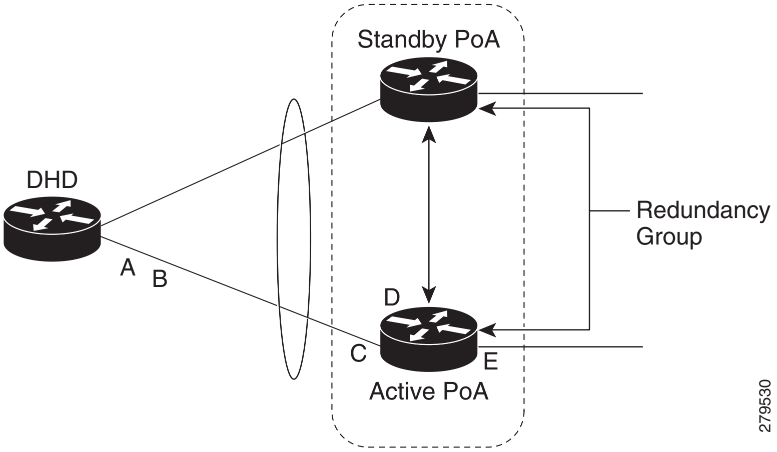
At any given time, only one POA is active for a bundle. Only the set of links between the DHD and the active POA actively sends traffic. The set of links between the DHD and the standby POA does not forward traffic. When the multichassis link bundle software detects that the connection to the active POA has failed, the software triggers the standby POA to become the active POA, and the traffic flows using the links between the DHD and newly active POA.

The ICCP protocol operates between the active and the standby POAs, and allows the POAs to coordinate their configuration, determine which POA is active, and trigger a POA to become active. Applications running on the two POAs (mLACP, IGMP snooping, DHCP snooping or ANCP) synchronize their state using ICCP.

Failure Modes

The mLACP feature provides network resiliency by protecting against port, link, and node failures. Figure 6 depicts the various failure modes.

Figure 6 Failure Modes



These are the failure categories:

- A—DHD uplink port failure. The port on the DHD that is connected to the POA fails.
- B—DHD uplink failure. The connection between the DHD and the POA fails.
- C—Active POA downlink port failure.
- D—Active POA node failure.
- E—Active POA uplink failure (network isolation). The links between the active POA and the core network fails

Core Network Redundancy Model

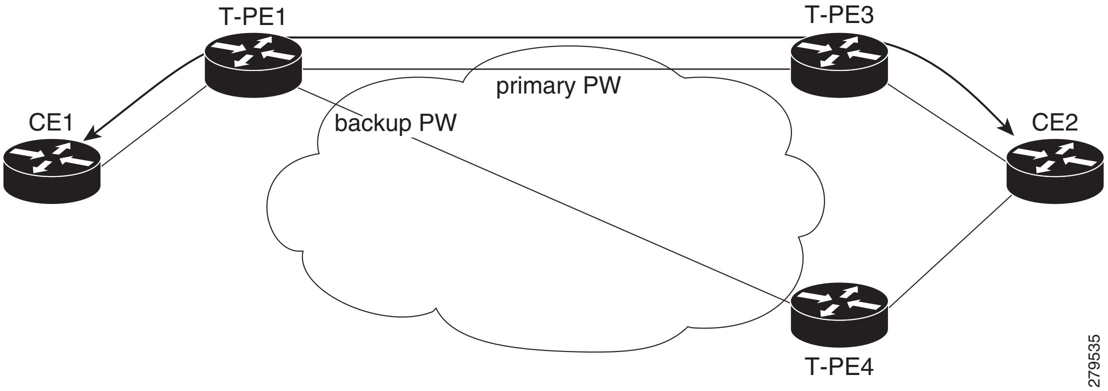
This section explains:

- One-way pseudowire redundancy
- Two-way pseudowire redundancy

One-way Pseudowire Redundancy

Figure 7 shows the VPWS one-way pseudowire redundancy model. Only one end of the pseudowire is protected by a backup pseudowire.

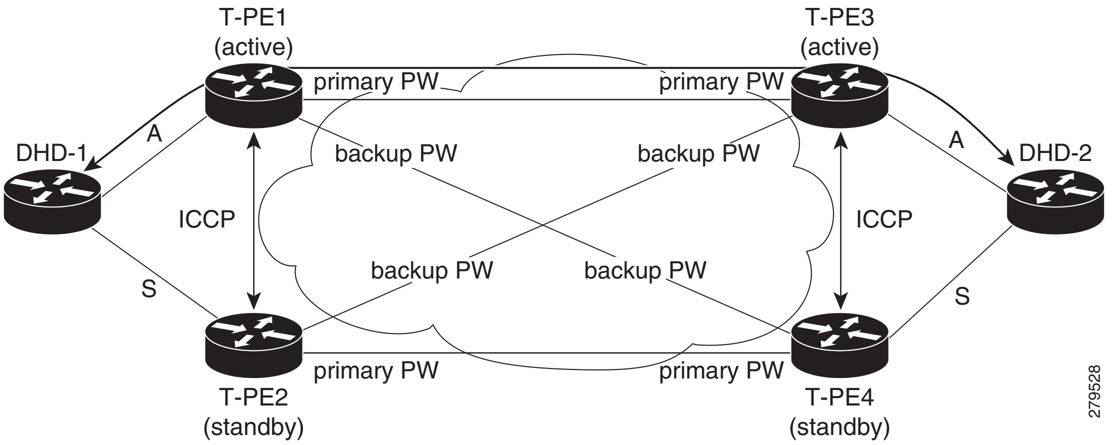
Figure 7 VPWS one-way Pseudowire Redundancy



Two-way Pseudowire Redundancy

Figure 8 shows the VPWS two-way pseudowire redundancy model. In this topology, each T-PE at the end of a PW has a primary and a backup PW. The state of the PW is coordinated with the state of the mLACP link between the DHD and the PE.

Figure 8 VPWS two-way Pseudowire Redundancy



Switchovers

Switchovers, which is changing the Active/Standby roles of the POAs, are performed using dynamic priority management or brute force behavior.

Dynamic Priority Management

Dynamic Priority Management involves co-ordination between the POAs to manipulate the LACP port priorities of their member links. Two priority values are tracked for each links:

- A configured priority which can either be configured explicitly, or defaults to 32768
- An operational priority used in LACP negotiations, which may differ from the configured priority if switchovers have occurred.

Higher priority LACP links are always selected ahead of lower priority LACP links. This means the operational priorities can be manipulated to force the standard LACP Selection Logic (on the POAs and on the DHD) to select desired links on both ends.

For example, consider a case where the DHD has two links to each POA, and each POA is configured with minimum-active links is 2. (This means the bundle goes down on the POA if the number of active links falls below 2.) The operational priorities for the member links are 1 on POA-1 and 2 on POA-2. This means that POA-1 is active (being higher priority) and the links on POA-2 are held in Standby state. The sequence of events in a switchover is as follows:

1. A link fails on POA-1, causing the number of active links to fall below the minimum of 2.
2. POA-1 changes the operational priority of both its links to 3, so the links on POA 2 are now higher priority.
3. POA-1 sends a LACP message to the DHD and an mLACP message to POA-2, informing both devices of the change.
4. The DHD tries to activate the links connected to POA-2 as these now have the highest priority.

5. POA-2 also ensures that its links have the highest priority and activates its links to the DHD.

At this point the switchover is complete.

Brute Force Behavior

In a brute force switchover, port priorities are not modified. Instead the failing POA sends a single *dying gasp* to the DHD over LACP, forcing it to deselect the link. It then terminates LACP communications on that link. This only leaves links between the DHD and POA-2, as links that can be selected. So, both ends select those links.

MC-LAG Topologies

This section illustrates the supported MC-LAG topologies.

Figure 9 VPWS One-way Pseudowire Redundancy in Redundancy Group

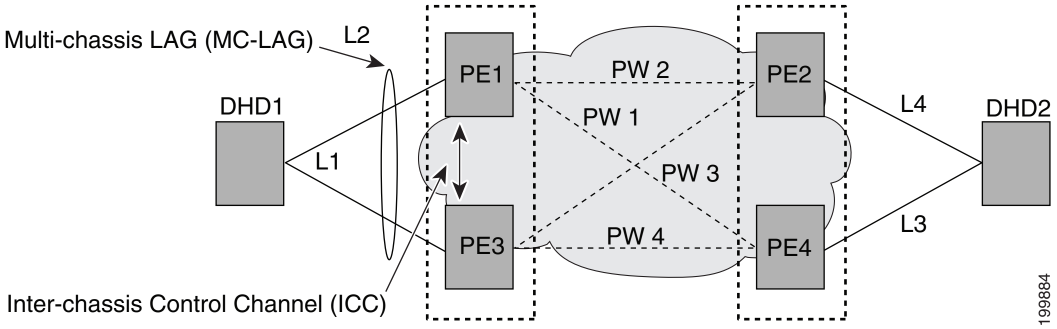

Multi-chassis LAG (MC-LAG)
L2
DHD1
L1
PE1
PE3
PW 2
PW 1
PW 3
PW 4
PE2
PE4
L4
L3
DHD2
Inter-chassis Control Channel (ICC)
199884

Figure 10 VPWS Two-way Pseudowire Redundancy

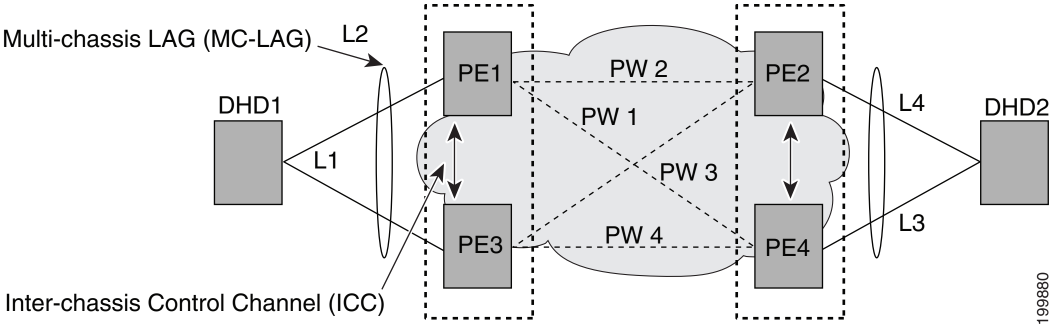

Multi-chassis LAG (MC-LAG)
L2
DHD1
L1
PE1
PE3
PW 2
PW 1
PW 3
PW 4
PE2
PE4
L4
L3
DHD2
Inter-chassis Control Channel (ICC)
199880

Figure 11 VPLS Pseudowires in One Redundancy Group

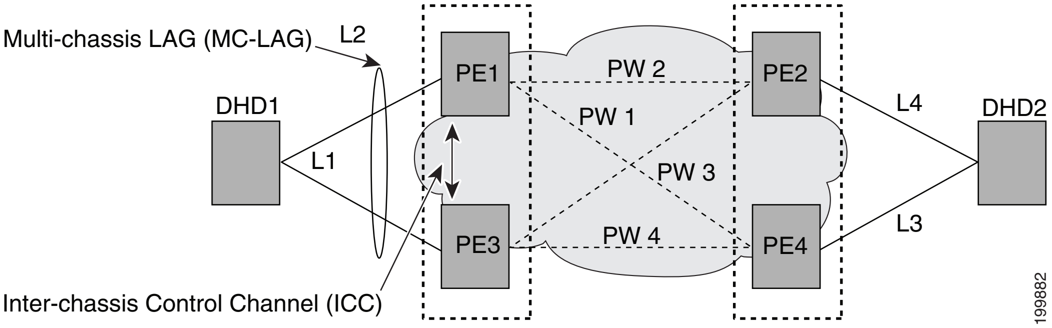

Multi-chassis LAG (MC-LAG)
L2
DHD1
L1
PE1
PE3
PW 2
PW 1
PW 3
PW 4
PE2
PE4
L4
L3
DHD2
Inter-chassis Control Channel (ICC)
199882

Figure 12 VPLS Pseudowires in Two Redundancy Groups

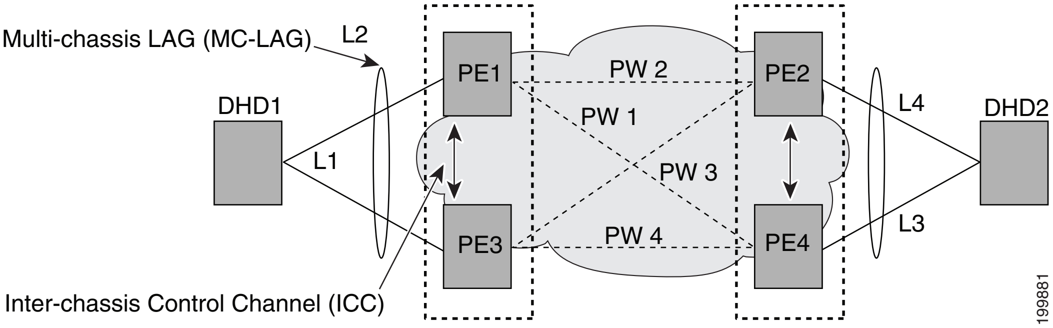

Multi-chassis LAG (MC-LAG)
L2
DHD1
L1
PE1
PE3
PW 2
PW 1
PW 3
PW 4
PE2
PE4
L4
L3
DHD2
Inter-chassis Control Channel (ICC)
199881

Figure 13 H-VPLS: EoMPLS over Access Pseudowire

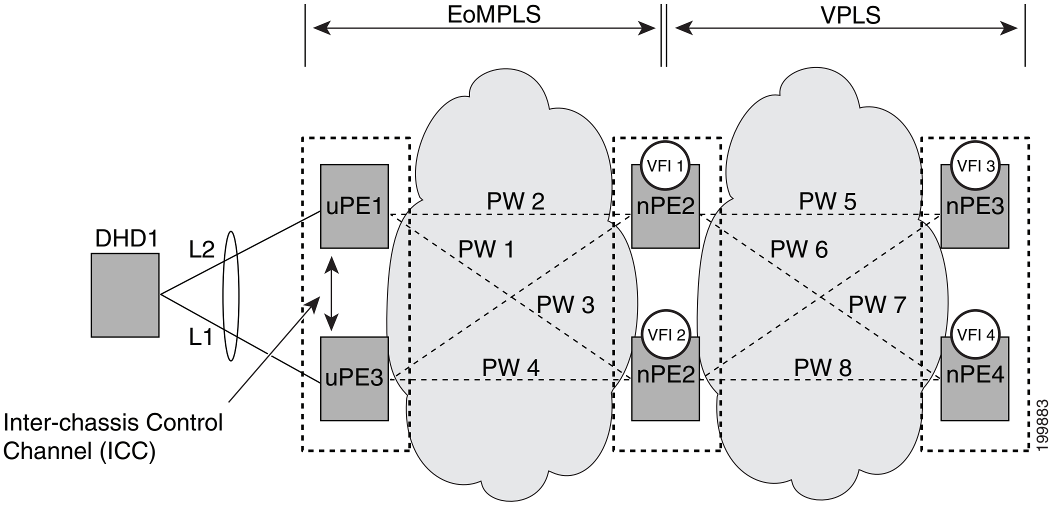
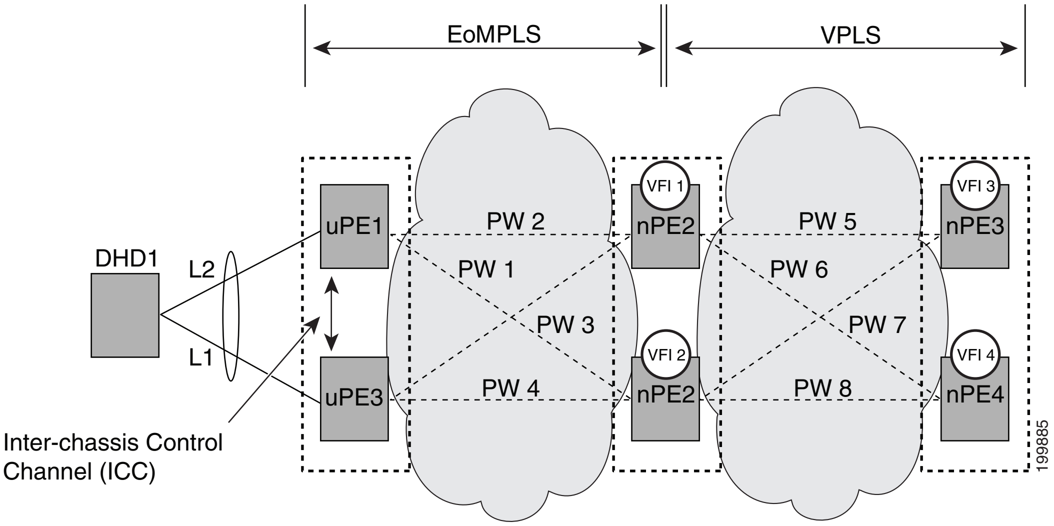


Figure 14 H-VPLS: VPWS Pseudowire on uPE and Access Pseudowire on nPE



Load Balancing

Load balancing is a forwarding mechanism that distributes traffic over multiple links based on certain parameters. The Cisco ASR 9000 Series Router supports load balancing for all links in a bundle using Layer 2, Layer 3, and Layer 4 routing information.

This section describes load balancing support on link bundles.

For more information about other forms of load balancing on the Cisco ASR 9000 Series Router, see the following references:

- Per-flow load balancing on non-bundle interfaces using Layer 3 and 4 routing information— See the *Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Configuration Guide*.
- Pseudowire (PW) Load Balancing beginning in Cisco IOS XR 4.0.1—See the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide.*

Layer 2 Ingress Load Balancing on Link Bundles

By default, load balancing on Layer 2 link bundles is done based on the MAC source and destination address (SA/DA) fields in the incoming packet header. Table 1 shows a summary of the parameters used for load balancing of incoming traffic at Layer 2 based on whether the default mode, EFP-based, or flow-based load balancing is in use.

Per-flow load balancing is supported on all links in the bundle. This scheme achieves load sharing by allowing the router to distribute packets over one of the links in the bundle, that is determined through a hash calculation. The hash calculation is an algorithm for link selection based on certain parameters.

The standard hash calculation is a 5-tuple hashing, using the following parameters:

- IP source address
- IP destination address
- Router ID
- Layer 4 source port

- Layer 4 destination port

When per-flow load balancing is enabled, all packets for a certain source-destination pair will go through the same link, though there are multiple links available. Per-flow load balancing ensures that packets for a certain source-destination pair arrive in order.

Note Load balancing for multicast traffic applies only when outgoing interfaces are link bundle interfaces or subinterfaces.

Table 1 Bundle Load Balancing for Incoming Traffic

Ingress Unicast, Flood, or Multicast Traffic	Parameters	Configuration
Default	• Source MAC address • Destination MAC address	n/a
EFP-based auto mode	XID of the xconnect	Auto mode is enabled using the **bundle load-balancing hash auto** command.
EFP-based with user hash	User hash	A user hash is configured using the **bundle load-balancing** *hash-value* command.
Flow-based with IP source and destination	• Source IP address • Destination IP address	Enabled using the L2VPN **load-balancing flow src-dst-ip** command.
Flow-based with MAC source and destination	• Source MAC address • Destination MAC address	Enabled using the L2VPN **load-balancing flow src-dst-mac** command.

Layer 3 Egress Load Balancing on Link Bundles

Layer 3 load balancing support began on the Cisco ASR 9000 Series Router in Cisco IOS XR 3.9.1, with changes introduced in Cisco IOS XR Release 4.0.1.

Layer 3 Load Balancing Before Cisco IOS XR Release 4.0.1

In Cisco IOS XR 3.9.1 through Cisco IOS XR 4.0, Layer 3 load balancing for link bundles is done on Ethernet Flow Points (EFPs) and is based on the IPv4 source and destination addresses in the packet. When Layer 3 service-specific load balancing is configured, all egressing bundles are load balanced based on the IPv4 source and destination addresses. When packets do not have IPv4 addresses, default load-balancing is used.

Layer 3 load balancing for link bundles is enabled globally, using the following command:

```
hw-module load-balance bundle l2-service l3-params
```

Layer 3 Load Balancing Beginning in Cisco IOS XR Release 4.0.1

Layer 3 load balancing for link bundles is done when outgoing interfaces are either bundles or bundle subinterfaces. 5-tuple hashing is used for load balancing among bundle member links, using the following parameters:

- IP source address
- IP destination address

- Router ID
- Layer 4 source port
- Layer 4 destination port

The ingress linecard does bundle member selection and forwards the packet to the linecard and network processor (NP) corresponding to the selected bundle member. The same hash value is used for both ingress and egress linecards. Therefore, even though the egress linecard also does bundle member selection, it selects the same bundle member that was selected by the ingress linecard.

Multicast IPv4 and IPv6 Traffic

For outbound multicast IPv4 or IPv6 traffic, a set of egress linecards is predetermined by the system. If a bundle interface or bundle subinterface is an outgoing interface, the system selects the bundle member for each outgoing interface in a route based on the multicast group address. This helps with load distribution of multicast routed traffic to different bundle members, while providing traffic sequencing within a specific route.

The egress linecard does NP selection using the same approach, when bundle members are spread across multiple NPs within the egress linecard.

When the packet arrives on an egress NP, it uses the 5-tuple hash to select a bundle member within an NP for each packet. This provides better resiliency for bundle member state changes within an NP.

Dynamic Load Balancing for LAG

Beginning in Cisco IOS XR Release 4.0.1, the Cisco ASR 9000 Series Router supports a method of dynamic load balancing among link aggregation (LAG) members. With dynamic load balancing, the hash algorithms for link selection include up to a maximum of 64 links, and are based on the current number of active members in the bundle.

QoS and Link Bundling

On the Cisco ASR 9000 Series Router, when QoS is applied on the bundle for either the ingress or egress direction, QoS is applied at each member interface. For complete information on configuring QoS on link bundles on the Cisco ASR 9000 Series Router, refer to the *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide* and the *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Command Reference*.

VLANs on an Ethernet Link Bundle

802.1Q VLAN subinterfaces can be configured on 802.3ad Ethernet link bundles. Keep the following information in mind when adding VLANs on an Ethernet link bundle:

- The maximum number of VLANs allowed per bundle is 4096.
- The maximum number of bundled VLANs allowed per router is 16384.

Note The memory requirement for bundle VLANs is slightly higher than standard physical interfaces.

To create a VLAN subinterface on a bundle, include the VLAN subinterface instance with the **interface Bundle-Ether** command, as follows:

interface Bundle-Ether *interface-bundle-id.subinterface*

After you create a VLAN on an Ethernet link bundle, all VLAN subinterface configuration is supported on that link bundle.

VLAN subinterfaces can support multiple Layer 2 frame types and services, such as Ethernet Flow Points - EFPs) and Layer 3 services.

Layer 2 EFPs are configured as follows:

```
interface bundle-ether instance.subinterface l2transport. encapsulation dot1q xxxxx
```

Layer 3 VLAN subinterfaces are configured as follows:

```
interface bundle-ether instance.subinterface, encapsulation dot1q xxxxx
```

Note The difference between the Layer 2 and Layer 3 interfaces is the **l2transport** keyword. Both types of interfaces use **dot1q encapsulation**.

Link Bundle Configuration Overview

The following steps provide a general overview of the link bundle configuration process. Keep in mind that a link must be cleared of all previous network layer configuration before it can be added to a bundle:

1. In global configuration mode, create a link bundle. To create an Ethernet link bundle, enter the **interface Bundle-Ether** command.
2. Assign an IP address and subnet mask to the virtual interface using the **ipv4 address** command.
3. Add interfaces to the bundle you created in Step 1 with the **bundle id** command in the interface configuration submode. You can add up to 64 links to a single bundle.

Note A link is configured as a member of a bundle from the interface configuration submode for that link.

Nonstop Forwarding During Card Failover

Cisco IOS XR software supports nonstop forwarding during failover between active and standby paired RSP cards. Nonstop forwarding ensures that there is no change in the state of the link bundles when a failover occurs.

For example, if an active RSP fails, the standby RSP becomes operational. The configuration, node state, and checkpoint data of the failed RSP are replicated to the standby RSP. The bundled interfaces will all be present when the standby RSP becomes the active RSP.

Note Failover is always onto the standby RSP.

Note You do not need to configure anything to guarantee that the standby interface configurations are maintained.

Link Failover

When one member link in a bundle fails, traffic is redirected to the remaining operational member links and traffic flow remains uninterrupted.

Multi-Gigabit Service Control Point

Multi-Gigabit Service Control Point (MGSCP) is a deployment model that uses certain link bundling and forwarding features on the Cisco ASR 9000 Series Aggregation Services Routers to support load balancing, clustering, and redundancy for broadband subscriber traffic on Cisco Service Control Engine (SCE) devices.

The Cisco SCE platform is used to provide many services such as user authorization, reporting, and application bandwidth metering for broadband subscribers. It manages IP traffic using a stateful processing mechanism based on application and subscriber awareness. Maintaining this statefulness requires that the SCE platform captures both the upstream and downstream flows of a session to classify it and provide Layer 7 processing at the application level.

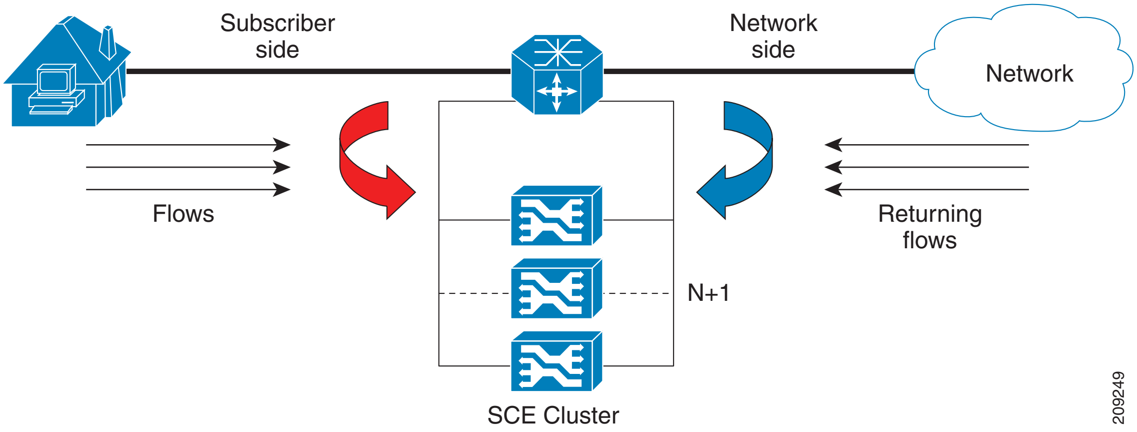
To process an application that is implemented with a bundle of flows, such as FTP or Session Initiation Protocol (SIP), the SCE platform needs to process all the flows that comprise a session of this application. In addition, when the SCE platform is configured to implement per subscriber reporting or control (sometimes referred to as *subscriber awareness*), it must process all traffic flows that a given subscriber generates.

Because of this stateful processing to the subscriber level, the SCE platform is implemented in a network with a “bump-in-the-wire” topology for Layer 2 and Layer 3 transparency. However, as the number of broadband subscribers increases along with the bandwidth that an SCE platform must support, scaling the solution presents certain challenges when inserted into a typical network environment where asymmetric routing is often implemented and the two directions of a single session, or the many flows of a specific subscriber, could be split between different links.

The MGSCP solution on the Cisco ASR 9000 Series Router satisfies these requirements by providing a topology to scale multiple SCE devices in a cluster that are connected to the router using link bundling, where all subscriber traffic can be directed through the same bundle member link. In addition, MGSCP also provides the benefits of load balancing and redundancy.

Figure 15 shows a basic network topology for MGSCP with a Cisco ASR 9000 Series Router connected between the subscriber and core networks, and acting as a dispatcher for the attached SCE cluster. The N+1 notation indicates one backup (or protect) link for the other active links on either side of the SCEs.

Figure 15 Basic MGSCP Network Topology



How to Configure Link Bundling

This section contains the following procedures:

- Configuring Ethernet Link Bundles, page 229
- Configuring EFP Load Balancing on an Ethernet Link Bundle, page 230
- Configuring VLAN Bundles, page 232
- Configuring POS Link Bundles, page 233
- Configuring Multichassis Link Aggregation, page 237

Configuring Ethernet Link Bundles

This section describes how to configure an Ethernet link bundle.

Note In order for an Ethernet bundle to be active, you must perform the same configuration on both connection endpoints of the bundle.

SUMMARY STEPS

The creation of an Ethernet link bundle involves creating a bundle and adding member interfaces to that bundle, as shown in the steps that follow.

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **ipv4 address** *ipv4-address mask*
4. **bundle minimum-active bandwidth** *kbps* (Optional)

5. **bundle minimum-active links** *links* (Optional)
6. **bundle maximum-active links** *links* (Optional)
7. **exit**
8. **interface** {**GigabitEthernet** | **TenGigE**}
9. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}
10. **no shutdown**
11. **exit**
12. Repeat Step 8 through Step 11 to add more links to the bundle you created in Step 2.
13. **end**
    or
    commit
14. **exit**
15. **exit**
16. Perform Step 1 through Step 15 on the remote end of the connection.
17. **show bundle Bundle-Ether** *bundle-id* [**reasons**]
18. **show lacp Bundle-Ether** *bundle-id*

Configuring EFP Load Balancing on an Ethernet Link Bundle

This section describes how to configure Ethernet flow point (EFP) Load Balancing on an Ethernet link bundle.

By default, Ethernet flow point (EFP) load balancing is enabled. However, the user can choose to configure all egressing traffic on the fixed members of a bundle to flow through the same physical member link. This configuration is available only on an Ethernet Bundle subinterface with Layer 2 transport (**l2transport**) enabled.

Note If the active members of the bundle change, the traffic for the bundle may get mapped to a different physical link that has a hash value that matches the configured value.

SUMMARY STEPS

Perform the following steps to configure EFP Load Balancing on an Ethernet link bundle:

1. **configure**
2. **hw-module load-balance bundle l2-service l3-params**
3. **interface Bundle-Ether** *bundle-id* **l2transport**
4. **bundle load-balance hash** *hash-value* [**auto**]
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**hw-module load-balance bundle l2-service l3-params** **Example:** `RP/0/RSP0/CPU0:router(config)# hw-module load-balance bundle l2-service l3-params`	(Optional) Enables Layer 3 load balancing on Layer 2 link bundles.
Step 3	**interface Bundle-Ether** *bundle-id* **l2transport** **Example:** `RP/0/RSP0/CPU0:router#(config)# interface Bundle-Ether 3 l2transport`	Creates a new Ethernet link bundle with the specified *bundle-id* and with Layer 2 transport enabled. The range is 1 to 65535.

Step 4	**bundle load-balance hash** *hash-value* [**auto**]
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# bundle load-balancing hash 1`
	or
	`RP/0/RSP0/CPU0:router(config-subif)# bundle load-balancing hash auto`
		Configures all egressing traffic on the fixed members of a bundle to flow through the same physical member link.
		• *hash-value*—Numeric value that specifies the physical member link through which all egressing traffic in this bundle will flow. The values are 1 through 8.
		• **auto**—The physical member link through which all egressing traffic on this bundle will flow is automatically chosen.

	Command or Action	Purpose

Step 5	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring VLAN Bundles

This section describes how to configure a VLAN bundle. The creation of a VLAN bundle involves three main tasks:

1. Create an Ethernet bundle.
2. Create VLAN subinterfaces and assign them to the Ethernet bundle.
3. Assign Ethernet links to the Ethernet bundle.

These tasks are describe in detail in the procedure that follows.

Note In order for a VLAN bundle to be active, you must perform the same configuration on both ends of the bundle connection.

SUMMARY STEPS

The creation of a VLAN link bundle is described in the steps that follow.

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **ipv4 address** *ipv4-address mask*
4. **bundle minimum-active bandwidth** *kbps* (Optional)

5. **bundle minimum-active links** *links* (Optional)
6. **bundle maximum-active links** *links* (Optional)
7. **exit**
8. **interface Bundle-Ether** *bundle-id.vlan-id*
9. **encapsulation dot1q**
10. **ipv4 address** *ipv4-address mask*
11. **no shutdown**
12. **exit**
13. Repeat Step 7 through Step 12 to add more VLANs to the bundle you created in Step 2.
14. **end**
    or
    commit
15. **exit**
16. **exit**
17. **show ethernet trunk bundle-Ether** *instance*
18. **configure**
19. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id*

Configuring POS Link Bundles

This section describes how to configure a POS link bundle.

Note In order for a POS bundle to be active, you must perform the same configuration on both connection endpoints of the POS bundle.

SUMMARY STEPS

The creation of a bundled POS interface involves configuring both the bundle and the member interfaces, as shown in these steps:

1. **configure**
2. **interface Bundle-POS** *bundle-id*
3. **ipv4 address** *ipv4-address mask*
4. **bundle minimum-active bandwidth** *kbps*
5. **bundle minimum-active links** *links*
6. **bundle maximum-active links** *links* [**hot-standby**]
7. **exit**
8. **interface POS** *interface-path-id*
9. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}]
10. **bundle port-priority** *priority*
11. **no shutdown**

12. **exit**
13. Repeat Step 19 through Step 21 to add more Ethernet interfaces to the bundle you created in Step 2.
14. **end**
    or
    commit
15. **exit**
16. **exit**
17. Perform Step 1 through Step 23 on the remote end of the connection.
18. **show bundle Bundle-POS** *bundle-id* [**reasons**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface Bundle-POS`** *`bundle-id`*
	Example:
	`RP/0/RSP0/CPU0:router#(config)#interface Bundle-POS 2`
		Configures and names the new bundled POS interface.
		Enters the interface configuration submode, from where interface specific configuration commands are executed. Use the **exit** command to exit from the interface configuration submode, and get back to the normal global configuration mode.

Step 3	**`ipv4 address`** *`ipv4-address mask`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.2.3 255.0.0.0`
		Assigns an IP address and subnet mask to the virtual interface using the ip address configuration subcommand.

Step 4	**`bundle minimum-active bandwidth`** *`kbps`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# bundle minimum-active bandwidth 620000`
		(Optional) Sets the minimum amount of bandwidth required before a user can bring up a bundle.

Step 5	**`bundle minimum-active links`** *`links`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# bundle minimum-active links 2`
		(Optional) Sets the number of active links required before you can bring up a specific bundle.

	Command or Action	Purpose
Step 6	**bundle maximum-active links** *links* [**hot-standby**] **Example:** `RP/0/RSP0/CPU0:router(config-if)# bundle maximum-active links 1 hot-standby`	(Optional) Implements 1:1 link protection for the bundle, which causes the highest-priority link in the bundle to become active and the second-highest-priority link to become the standby. Also, specifies that a switchover between active and standby LACP-enabled links is implemented according to a proprietary optimization. **Note** The priority of the active and standby links is based on the value of the **bundle port-priority** command.
Step 7	**exit**	Exits the interface configuration submode.
Step 8	**interface POS** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface POS 0/1/0/0`	Enters POS interface configuration mode and specifies the POS interface name and interface-path-id notation *rack/slot/module/port*.
Step 9	**bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}] **Example:** `RP/0/RSP0/CPU0:router(config-if)# bundle-id 3`	Adds the link to the specified bundle. To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string. To add the link to the bundle without LACP support, include the optional **mode on** keywords with the command string. **Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port).
Step 10	**bundle port-priority** *priority* **Example:** `RP/0/RSP0/CPU0:router(config-if)# bundle port-priority 1`	(Optional) If you set the **bundle maximum-active links** command to 1, you must also set the priority of the active link to the highest priority (lowest value) and the standby link to the second-highest priority (next lowest value). For example, you can set the priority of the active link to 1 and the standby link to 2.
Step 11	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration which forces the interface administratively down. The **no shutdown** command then returns the link to an up or down state, depending on the configuration and state of the link.
Step 12	**exit** **Example:** `RP/0/RSP0/CPU0:router# exit`	Exits the interface configuration submode for the POS interface.
Step 13	Repeat Step 19 through Step 21 to add more links to a bundle	(Optional) Adds more links to the bundle you created in Step 2.

	Command or Action	Purpose

Step 14	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 15	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits interface configuration mode.

Step 16	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# exit`
		Exits global configuration mode.

Step 17	Perform Step 1 through Step 23 on the remote end of the connection.	Brings up the other end of the link bundle.

Step 18	**`show bundle Bundle-POS`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router# show bundle Bundle-POS 1`
		(Optional) Shows information about the specified POS link bundle.

Configuring Multichassis Link Aggregation

Perform these tasks to configure Multichassis Link Aggregation (MC-LAG):

- Configuring Interchassis Communication Protocol, page 237
- Configuring Multichassis Link Aggregation Control Protocol Session, page 240
- Configuring Multichassis Link Aggregation Control Protocol Bundle, page 242
- Configuring Dual-Homed Device, page 244
- Configuring Access Backup Pseudowire, page 246
- Configuring One-way Pseudowire Redundancy in MC-LAG, page 249
- Configuring VPWS Cross-Connects in MC-LAG, page 251
- Configuring VPLS in MC-LAG, page 254

Configuring Interchassis Communication Protocol

Perform this task to configure Interchassis Communication Protocol (ICCP).

SUMMARY STEPS

1. **configure**
2. **redundancy iccp group** *group-id*
3. **member neighbor** *neighbor-ip-address*
4. **backbone interface** *interface-type-id*
5. **isolation recovery-delay** *delay*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`redundancy iccp group`** *`group-id`*
	`Example:`
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group)`
	`# redundancy iccp group 100`
		Adds an ICCP redundancy group.

	Command or Action	Purpose

Step 3	**`member neighbor`** *`neighbor-ip-address`*
	`Example:`
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group) # member neighbor 10.1.1.1`
		Configures ICCP members.
		This is the ICCP peer for this redundancy group. Only one neighbor can be configured per redundancy group. The IP address is the LDP router-ID of the neighbor. This configuration is required for ICCP to function.

Step 4	**`backbone interface`** *`interface-type-id`*
	`Example:`
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group) # backbone interface GigabitEthernet0/1/0/2`
		Configures ICCP backbone interfaces.
		This is an optional configuration to detect isolation from the network core, and triggers switchover to the peer POA if the POA on which the failure is occurring is active. Multiple backbone interfaces can be configured for each redundancy group. When all backbone interfaces are not UP, this is an indication of core isolation. When one or more backbone interfaces are UP, then the POA is not isolated from the network core. Backbone interfaces are typically the interfaces which L2VPN pseudowires can use.

	Command or Action	Purpose

Step 5	**`isolation recovery-delay`** *`delay`*
	
	`Example:`
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group) # isolation recovery-delay 30`
		Configures the isolation parameters and specifies delay before clearing isolation condition after recovery from failure.
		Isolation recovery delay timer is started once the core isolation condition has cleared. When the timer expires, the POA can take over as the active POA (depending on other conditions like bundle recovery delay timer). This allows:
		• the network core to reconverge after the backbone interfaces have come up
		• ICCP state to be exchanged in order for POAs to know what state they are supposed to be in so that MCLAG bundles do not flap excessively.
		This is an optional configuration; if not configured, the delay is set to 180 seconds, by default.

Step 6	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-redundancy-iccp-group)# end`
	or
	`RP/0/RSP0/CPU0:router(config-redundancy-iccp-group)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Multichassis Link Aggregation Control Protocol Session

Perform this task to enable a Multichassis Link Aggregation Control Protocol (mLACP) session.

SUMMARY STEPS

1. **configure**
2. **redundancy iccp group** *group-id*
3. **mlacp system mac** *mac-id*
4. **mlacp system priority** *priority*
5. **mlacp node** *node-id*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`redundancy iccp group`** *`group-id`*
	`Example:`
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group)`
	`# redundancy iccp group 100`
		Adds an ICCP redundancy group.

Step 3	**`mlacp system mac`** *`mac-id`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group)`
	`# mlacp system mac 1.1.1`
		Configures the LACP system ID to be used in this ICCP Group.
		Note The *mac-id* is a user configured value for the LACP system LAG-ID to be used by the POAs. It is highly recommended that the *mac-ids* have the same value on both POAs. You can have different LAG-IDs for different groups.

Step 4	**`mlacp system priority`** *`priority`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group)`
	`# mlacp system priority 10`
		Sets the LACP system priority to be used in this ICCP Group.
		Note It is recommended that system priority of the POAs be configured to a lower numerical value (higher priority) than the LACP LAG ID of the DHD. If the DHD has higher system priority then dynamic priority management cannot work and brute force switchover is automatically used.

	Command or Action	Purpose

Step 5	**`mlacp node`** *`node-id`*
	
	Example:
	`RP/0/RSP0/CPU0:router#(config-redundancy-iccp-group)`
	`# mlacp node 1`
		Sets the LACP system priority to be used in this ICCP Group.
		Note The *node-id* must be unique for each POA.

Step 6	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Multichassis Link Aggregation Control Protocol Bundle

Perform this task to configure a Multichassis Link Aggregation Control Protocol (mLACP) bundle.

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **mac-address** *mac-id*
4. **bundle wait-while** *milliseconds*
5. **lacp switchover suppress-flaps** *milliseconds*
6. **mlacp iccp-group** *group-id*
7. **mlacp port-priority** *priority*
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface Bundle-Ether`** *`bundle-id`*
	Example:
	`RP/0/RSP0/CPU0:router#(config)# interface Bundle-Ether 3`
		Creates and names a new Ethernet link bundle.

Step 3	**`mac-address`** *`mac-id`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-if)# mac-address 1.1.1`
		Sets the MAC address on the interface.
		Note Configuring the same MAC address on both POAs is highly recommended.

Step 4	**`bundle wait-while`** *`milliseconds`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-if)# bundle wait-while 100`
		Sets the wait-while timeout for members of this bundle.

Step 5	**`lacp switchover suppress-flaps`** *`milliseconds`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-if)# lacp switchover suppress-flaps 300`
		Sets the time for which to suppress flaps during a LACP switchover.
		Note It is recommended that the value used for the *milliseconds* argument is greater than that for the wait-while timer of the local device (and DHD).

	Command or Action	Purpose

Step 6	**`mlacp iccp-group`** *`group-id`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-if)# mlacp iccp-group 10`
		Configures the ICCP redundancy group in which this bundle should operate.

Step 7	**`mlacp port-priority`** *`priority`*
	Example:
	`RP/0/RSP0/CPU0:router#(config-if)# mlacp port-priority 10`
		Sets the starting priority for all member links on this device when running mLACP.
		Note Lower value indicates higher priority. If you are using dynamic priority management the priority of the links change when switchovers occur.

Step 8	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Dual-Homed Device

Perform this task to configure the dual-homed device (DHD).

Note If an ASR 9000 Series Router is to be used as a DHD, it is recommended that you configure the **bundle maximum-active links** *links* command where *links* is the number of links connecting the DHD to one of the POAs.

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bundle wait-while** *milliseconds*
4. **lacp switchover suppress-flaps** *milliseconds*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RSP0/CPU0:router#(config-if)# interface Bundle-Ether 3`	Creates and names a new Ethernet link bundle.
Step 3	**bundle wait-while** *milliseconds* **Example:** `RP/0/RSP0/CPU0:router#(config-if)# bundle wait-while 100`	Sets the wait-while timeout for members of this bundle.

	Command or Action	Purpose
Step 4	**lacp switchover suppress-flaps** *milliseconds* **Example:** `RP/0/RSP0/CPU0:router#(config-if)# lacp switchover suppress-flaps 300`	Sets the time for which to suppress flaps during a LACP switchover.

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

The members added to the bundle on one POA go *Active*, and the members on the other POA are in *Standby* state. This can be verified by using the **show bundle** command on either POA to display the membership information for correctly configured members on both the POAs:

```
RP/0/RSP0/CPU0:router# show bundle

Bundle-Ether1
  Status:                                    Up
  Local links <active/standby/configured>:   1 / 0 / 1
  Local bandwidth <effective/available>:     1000000 (1000000) kbps
  MAC address (source):                      0000.deaf.0000 (Configured)
  Minimum active links / bandwidth:          1 / 1 kbps
  Maximum active links:                      64
  Wait while timer:                          100 ms
  LACP:                                      Operational
    Flap suppression timer:                  300 ms
  mLACP:                                     Operational
    ICCP Group:                              1
    Role:                                    Active
    Foreign links <active/configured>:       0 / 1
    Switchover type:                         Non-revertive
    Recovery delay:                          300 s
    Maximize threshold:                      Not configured
  IPv4 BFD:                                  Not configured
```

```
Port                  Device          State       Port ID         B/W, kbps
--------------------  --------------- ----------- --------------- ----------
Gi0/0/0/0             Local           Active      0x8001, 0x9001     1000000
    Link is Active
Gi0/0/0/0             5.4.3.2         Standby     0x8002, 0xa001     1000000
    Link is marked as Standby by mLACP peer
```

Note To switch to an active POA, use the **mlacp switchover Bundle-Ether** command on the currently active router.

Configuring Access Backup Pseudowire

Perform this task to add a backup pseudowire to a VPLS Access pseudowire.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge-group name*
4. **bridge-domain** *bridge-domain name*
5. **neighbor** *A.B.C.D ip-address* **pw-id** *pseudowire-id*
6. **pw-class** {*class-class name*}
7. **backup neighbor** *A.B.C.D ip-address* **pw-id** *pseudowire-id*
8. **pw-class** {*class-class name*}
9. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	Example:
	`RP/0/RSP0/CPU0:router(config)# l2vpn`
	`RP/0/RSP0/CPU0:router(config-l2vpn)#`
		Enters L2VPN configuration mode.

Step 3	**bridge group** *bridge-group-name*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

	Command or Action	Purpose

Step 4	**bridge-domain** *bridge-domain-name*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg)#`
	`bridge-domain abc`
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**neighbor** *A.B.C.D* **pw-id** *pseudowire-id*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# neighbor`
	`10.2.2.2 pw-id 2000`
		Configures the pseudowire segment.

Step 6	**pw-class** {*class-name*}
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)#`
	`pw-class class1`
		Configures the pseudowire class template name to use for the pseudowire.

Step 7	**backup neighbor** *A.B.C.D* **pw-id** *pseudowire-id*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# backup`
	`neighbor 10.2.2.2 pw-id 2000`
		Adds a backup pseudowire to a VPLS access pseudowire (PW).

	Command or Action	Purpose

Step 8	**`pw-class`** {*`class-name`*}
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# pw-class class2`
		Configures the pseudowire class template name to use for the backup pseudowire.

Step 9	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring One-way Pseudowire Redundancy in MC-LAG

Perform this task to allow one-way pseudowire redundancy behavior when the redundancy group is configured.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **pw-class** {*class-name*}
4. **encapsulation mpls**
5. **redundancy one-way**
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# l2vpn`
	`RP/0/RSP0/CPU0:router(config-l2vpn)#`
		Enters L2VPN configuration mode.

Step 3	**`pw-class`** {*`class-name`*}
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# pw-class class1`
		Configures the pseudowire class template name to use for the pseudowire.

Step 4	**`encapsulation mpls`**
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls`
		Configures the pseudowire encapsulation to MPLS.

	Command or Action	Purpose

Step 5	**`redundancy one-way`**
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mpls)# redundancy one-way`
		Configures one-way PW redundancy behavior.
		Note The **redundancy one-way** command is effective only if the redundancy group is configured.

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring VPWS Cross-Connects in MC-LAG

Perform this task to configure VPWS cross-connects in MC-LAG.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **pw-status**
4. **xconnect group** *group-name*
5. **p2p** *xconnect-name*
6. **interface** *type interface-path-id*
7. **neighbor** *A.B.C.D ip-address* **pw-id** *pseudowire-id*
8. **pw-class** {*class-class name*}
9. **backup neighbor** *A.B.C.D ip-address* **pw-id** *pseudowire-id*
10. **pw-class** {*class-class name*}
11. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# l2vpn`
		Enters L2VPN configuration mode.

Step 3	**`pw-status`**
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# pw-status`
		Enables pseudowire status.
		Note When the attachment circuit changes redundancy state to Active, Active pw-status is sent over the primary and backup pseudowires.
		When the attachment circuit changes redundancy state to Standby, Standby pw-status is sent over the primary and backup pseudowires.

	Command or Action	Purpose
Step 4	**xconnect group** *group-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group grp_1`	Enters the name of the cross-connect group.
Step 5	**p2p** *xconnect-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p p1`	Enters a name for the point-to-point cross-connect.
Step 6	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface Bundle-Ether 1.1`	Specifies the interface type ID.
Step 7	**neighbor** *A.B.C.D* **pw-id** *pseudowire-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor 10.2.2.2 pw-id 2000`	Configures the pseudowire segment for the cross-connect. Optionally, you can disable the control word or set the transport-type to Ethernet or VLAN.

Step 8	**pw-class** {*class-name*}
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# pw-class c1`
		Configures the pseudowire class template name to use for the pseudowire.

Step 9	**backup neighbor** *A.B.C.D* **pw-id** *pseudowire-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# backup neighbor 10.2.2.2 pw-id 2000`	Adds a backup pseudowire.

	Command or Action	Purpose

Step 10	**`pw-class`** `{`*`class-name`*`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw-backup) # pw-class c2`
		Configures the pseudowire class template name to use for the backup pseudowire.

Step 11	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw-backup) # end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw-backup) # commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring VPLS in MC-LAG

Perform this task to configure VPLS in MC-LAG.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **pw-status**
4. **bridge group** *bridge-group-name*
5. **bridge-domain** *bridge-domain-name*
6. **interface** *type interface-path-id*
7. **vfi** *vfi-name*
8. **neighbor** *A.B.C.D ip-address* **pw-id** *pseudowire-id*
9. **pw-class** {*class-class name*}
10. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**l2vpn** **Example:** `RP/0/RSP0/CPU0:router(config)# l2vpn`	Enters L2VPN configuration mode.
Step 3	**pw-status** **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn)# pw-status`	(Optional) Enables pseudowire status. All the pseudowires in the VFI are always active, independent of the attachment circuit redundancy state.

Step 4	**bridge group** *bridge-group-name*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

	Command or Action	Purpose

Step 5	**bridge-domain** *bridge-domain-name*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

Step 6	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface Bundle-Ether 1.1`	Specifies the interface type ID.

Step 7	**vfi** *{vfi-name}*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# vfi vfi-east`
		Enters virtual forwarding instance (VFI) configuration mode.

Step 8	**neighbor** *A.B.C.D* **pw-id** *pseudowire-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.2.2.2 pw-id 2000`	Configures the pseudowire segment for the cross-connect. Optionally, you can disable the control word or set the transport-type to Ethernet or VLAN.

	Command or Action	Purpose

Step 9	**pw-class** {*class-name*}
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# pw-class canada`
		Configures the pseudowire class template name to use for the pseudowire.

Step 10	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

How to Configure MGSCP

- Prerequisites for Configuring MGSCP, page 256
- Restrictions for Configuring MGSCP, page 257
- Configuring the Access Bundle for the Subscriber-Facing Side, page 258 (required)
- Configuring the Network Bundle for the Core-Facing Side, page 260 (required)
- Configuring the Bundle Member Interfaces, page 262 (required)
- Configuring VRFs to Route Traffic to the Bundles, page 263 (recommended)

Prerequisites for Configuring MGSCP

Before configuring MGSCP, be sure that the following prerequisites are met:

- You have Gigabit Ethernet or 10-Gigabit Ethernet line cards installed in the Cisco ASR 9000 Series Router.

- You understand how to configure your cluster of Service Control Engine (SCE) devices and configure them according to the desired requirements of your network, including the following requirements for MGSCP support:
  - When you connect the SCE devices to the Cisco ASR 9000 Series Router, be sure that each SCE device has two separate physical links connecting to two different bundle interfaces on the Cisco ASR 9000 Series Router as follows:

    — One link from each SCE device is connected to a link on the bundle interface that is routed to the access (or subscriber) side of the network.

    — The second link from each SCE device is connected to a link on another bundle interface that is routed to the core side of the network.
  - On the SCE device, you configure the SCE ports for link failure reflection (using the **link failure-reflection** command) to ensure that if a link on one side of the SCE goes down, then the link on the other side is automatically shut down. For more information, see the "Configuring the Connection" chapter in the Cisco SCE software configuration guide for your device and release at:

    http://www.cisco.com/en/US/products/ps6134/products_installation_and_configuration_guides_list.html
- For your bundle configuration on the Cisco ASR 9000 Series Router, determine the following information:
  - The maximum number of active links that you will support.
  - The bundle links that will be protect (backup) links. You can configure a maximum of 4 protect links.
- To maintain the statefulness of the connected SCEs, all subscriber flows must pass through the same SCE. Therefore, before you configure MGSCP, you need to determine how you want to configure the router to redirect subscriber traffic to ensure that it passes through the appropriate bundle interfaces connected to that SCE.

  You can use one of the following methods:
  - ACL-Based Forwarding (ABF)—Supports only IP addresses for the next hop, and can be complex to configure. For more information about ABF, see the "Implementing Access Lists and Prefix Lists" chapter of the *Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Configuration Guide*.
  - Virtual Routing and Forwarding (VRF)—Recommended. Uses VRF instances for the access and network bundles, which can then be routed using static or dynamic routing with OSPF and BGP.

Restrictions for Configuring MGSCP

Before configuring MGSCP, consider these restrictions:

- You can configure up to a maximum of 4 protect links on a bundle.
- IPv6 addressing is not supported. IPv4 addressing must be used.
- MPLS is not supported.
- You can configure up to a maximum of 8 member links on a bundle.

Configuring the Access Bundle for the Subscriber-Facing Side

The configuration of the access bundle facing the subscriber side of the network is similar to the core bundle configuration, with the following guidelines:

- If using VRFs to route subscriber traffic on the same SCE to the bundle (recommended), then a separate VRF is used for the subscriber-facing side.
- Link-order signaling is required to enable LACP processing of link ordering numbers (LONs) for load balancing tables.
- Bundle load balancing is configured based on source IP address.
- The maximum number of active links must be configured to match the maximum number of active links on the core bundle.

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **vrf** *vrf-name*
4. **ipv4 address** *ipv4-address mask*
5. **lacp cisco enable link-order signaled**
6. **bundle load-balancing hash src-ip**
7. **bundle maximum-active links** *links* [**hot-standby**]
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface Bundle-Ether`** *`bundle-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether 100`
		Specifies or creates an Ethernet bundle interface for the subscriber-facing side of the network, where *bundle-id* is a number from 1 to 65535, and enters interface configuration mode.

Step 3	**`vrf`** *`vrf-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# vrf access`
		(Optional—Recommended) Specifies the VRF instance for the subscriber-facing side of the network in which this Ethernet bundle participates.

	Command or Action	Purpose

Step 4	**`ipv4 address`** *`ipv4-address mask`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.1.1 255.255.255.0`
		Specifies an IPv4 address and mask that is part of the specified VRF for this interface, where *ipv4-address* is the 32-bit IP address wtih corresponding mask in dotted-decimal format (A.B.C.D).
		Note This command must be specified after the **vrf** command to be sure that the IP address is part of the VRF instance.

Step 5	**`lacp cisco enable link-order signaled`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# lacp cisco enable link-order signaled`
		Enables the use of Cisco TLVs to include link order numbering as part of the LACP processing on this bundle.

Step 6	**`bundle load-balancing hash src-ip`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# bundle load-balancing hash src-ip`
		Specifies that the hash used for load balancing on the subscriber bundle interface is based on source IP address.

Step 7	**`bundle maximum-active links`** *`links`* [**`hot-standby`**] **Example:** `RP/0/RSP0/CPU0:router(config-if)# bundle maximum-active links 2`	Specifies the maximum number of active links allowed for the bundle, and sets the upper bound on the link ordering numbers in use for load balancing tables. **Note** To support MGSCP, this command must also be configured with the same value on the core bundle.

Step 8	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router (config-bfd-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-bfd-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Network Bundle for the Core-Facing Side

The configuration of the bundle facing the core side of the network is similar to the access bundle configuration, with the following guidelines:

- If using VRFs to route subscriber traffic on the same SCE to the bundle (recommended), then a separate VRF is used for the core-facing side.
- Link-order signaling is required to enable LACP processing of LONs for load balancing tables.
- Bundle load balancing is configured based on destination IP address.
- The maximum number of active links must be configured to match the maximum number of active links on the access bundle.

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **vrf** *vrf-name*
4. **ipv4 address** *ipv4-address mask*
5. **lacp cisco enable link-order signaled**
6. **bundle load-balancing hash dst-ip**
7. **bundle maximum-active links** *links* [**hot-standby**]
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether 100`	Specifies or creates an Ethernet bundle interface for the subscriber-facing side of the network, where *bundle-id* is a number from 1 to 65535, and enters interface configuration mode.
Step 3	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-if)# vrf access`	(Optional—Recommended) Specifies the VRF instance for the core-facing side of the network in which this Ethernet bundle participates.

	Command or Action	Purpose
Step 4	`ipv4 address` *ipv4-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.1.1 255.255.255.0`	Specifies an IPv4 address and mask that is part of the specified VRF for this interface, where *ipv4-address* is the 32-bit IP address wtih corresponding mask in dotted-decimal format (A.B.C.D). **Note** This command must be specified after the **vrf** command to be sure that the IP address is part of the VRF instance.

Step 5	`lacp cisco enable link-order signaled`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# lacp cisco enable link-order signaled`
		Enables the use of Cisco TLVs to include link order numbering as part of the LACP processing on this bundle.

Step 6	`bundle load-balancing hash dst-ip`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# bundle load-balancing hash dst-ip`
		Specifies that the hash used for load balancing on the subscriber bundle interface is based on destination IP address.

Step 7	`bundle maximum-active links` *links* [**hot-standby**] **Example:** `RP/0/RSP0/CPU0:router(config-if)# bundle maximum-active links 2`	Specifies the maximum number of active links allowed for the bundle, and sets the upper bound on the link ordering numbers in use for load balancing tables. **Note** To support MGSCP, this command must also be configured with the same value on the access bundle.

Step 8	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router (config-bfd-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-bfd-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Bundle Member Interfaces

When the access and core bundles have been configured, bundle interfaces must be configured as the active and protect links on those bundles, with the following guidelines:

- A link becomes a bundle member using the **bundle id** command and specifying the ID of the corresponding bundle interface. For MGSCP, there are two different bundles: one for the access side traffic, and one for the core side traffic. These bundles each have a link connecting to either side of an SCE. Be sure to carefully map your interfaces to the appropriate bundle.
- LACP is required for MGSCP, so the link must be configured with **mode active** on the bundle.
- Active and backup (protect) links are configured using the **bundle port-priority** command:
  - To configure a working (active) link, use a priority of 1. The maximum number of active links that you can configure is determined by the value of the **bundle maximum-active links** command on the bundle.
  - Any priority other than 1 designates the link as a protect link. You can configure a maximum of 4 protect links.

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **bundle id** *bundle-id* **mode active**
4. **bundle port-priority** *priority*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`**`]` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0`
		Specifies or creates a Gigabit Ethernet or 10-Gigabit Ethernet interface, where *interface-path-id* is the physical location of the interface using *rack***/***slot***/***module***/***port* notation, and enters interface configuration mode.

Step 3	**`bundle id`** *`bundle-id`* **`mode active`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# bundle id 100 mode active`
		Adds the interface as a member of the specified bundle, and runs LACP in active mode on the interface to exchange LACP packets for MGSCP.

	Command or Action	Purpose
Step 4	**bundle port-priority** *priority* **Example:** `RP/0/RSP0/CPU0:router(config-if)# bundle port-priority 1`	Specifies the LACP priority for the interface and determines if a bundle interface is an active or protect link for MGSCP: • Value of 1—Specifies the link is an active interface. • Value other than 1—Specfies the link is a protect interface. The default is 32768.

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router (config-bfd-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-bfd-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring VRFs to Route Traffic to the Bundles

VRFs are the recommended way to route subscriber traffic to the bundles to be sure that all subscriber traffic remains with the same SCE device for statefulness. To configure VRFs for MGSCP, complete one of the following tasks:

- Configuring VRFs with Static Routing, page 263
- Configuring VRFs with Dynamic Routing, page 264

Configuring VRFs with Static Routing

These steps summarize the tasks required to configure VRFs using static routing:

1. Configure two VRFs in global configuration—one each for the access and core sides of the network. Be sure to specify the IPv4 unicast address family.
2. Configure IPv4 addresses at each of the bundle interfaces and associate those addresses with the corresponding VRF that you configured in global configuration for the access and core side of the network.

3. Configure IPv4 addresses at the Gigabit Ethernet physical interfaces and associate those addresses with the corresponding VRF that you configured in global configuration for the access and core side of the network.
4. Configure static routing using the **router static** command to map the access and core VRFs to their corresponding bundle interfaces.

For a sample configuration, see the "Example: Configuring VRFs with Static Routing" section on page 272.

Configuring VRFs with Dynamic Routing

VRFs for MGSCP are supported for both OSPF and BGP routing protocols. The general configuration of the VRFs in global configuration and at the bundle and physical interfaces is the same as for static routing.

These steps summarize the tasks required to configure VRFs using OSPF routing:

1. Configure two VRFs in global configuration—one each for the access and core sides of the network. Be sure to specify the IPv4 unicast address family.
2. Configure IPv4 addresses at each of the bundle interfaces and associate those addresses with the corresponding VRF that you configured in global configuration for the access and core side of the network.
3. Configure IPv4 addresses at the Gigabit Ethernet physical interfaces and associate those addresses with the corresponding VRF that you configured in global configuration for the access and core side of the network.
4. Configure a dynamic routing protocol, such as OSPF, using the **router ospf** command to define the VRFs and associate the bundle and physical interfaces to the OSPF areas.

For a sample configuration, see the "Example: Configuring VRFs with OSPF Routing" section on page 273

Configuration Examples for Link Bundling

This section contains the following examples:

- Example: Configuring an Ethernet Link Bundle, page 264
- Example: Configuring a VLAN Link Bundle, page 265
- Example: Configuring a POS Link Bundle, page 265
- Example: Configuring EFP Load Balancing on an Ethernet Link Bundle, page 266
- Example: Configuring Multichassis Link Aggregation, page 266

Example: Configuring an Ethernet Link Bundle

The following example shows how to join two ports to form an EtherChannel bundle running LACP:

```
RP/0/RSP0/CPU0:Router# config
RP/0/RSP0/CPU0:Router(config)# interface Bundle-Ether 3
RP/0/RSP0/CPU0:Router(config-if)# ipv4 address 1.2.3.4/24
RP/0/RSP0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/RSP0/CPU0:Router(config-if)# bundle minimum-active links 1
RP/0/RSP0/CPU0:Router(config-if)# exit
RP/0/RSP0/CPU0:Router(config)# interface TenGigE 0/3/0/0
RP/0/RSP0/CPU0:Router(config-if)# bundle id 3 mode active
RP/0/RSP0/CPU0:Router(config-if)# no shutdown
RP/0/RSP0/CPU0:Router(config)# exit
RP/0/RSP0/CPU0:Router(config)# interface TenGigE 0/3/0/1
RP/0/RSP0/CPU0:Router(config-if)# bundle id 3 mode active
RP/0/RSP0/CPU0:Router(config-if)# no shutdown
RP/0/RSP0/CPU0:Router(config-if)# exit
```

Example: Configuring a VLAN Link Bundle

The following example shows how to create and bring up two VLANs on an Ethernet bundle:

```
RP/0/RSP0/CPU0:Router# config
RP/0/RSP0/CPU0:Router(config)# interface Bundle-Ether 1
RP/0/RSP0/CPU0:Router(config-if)# ipv4 address 1.2.3.4/24
RP/0/RSP0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/RSP0/CPU0:Router(config-if)# bundle minimum-active links 1
RP/0/RSP0/CPU0:Router(config-if)# exit
RP/0/RSP0/CPU0:Router(config)# interface Bundle-Ether 1.1
RP/0/RSP0/CPU0:Router(config-subif)# encapsulation dot1q 10
RP/0/RSP0/CPU0:Router(config-subif)# ip addr 10.2.3.4/24
RP/0/RSP0/CPU0:Router(config-subif)# no shutdown
RP/0/RSP0/CPU0:Router(config-subif)# exit
RP/0/RSP0/CPU0:Router(config)# interface Bundle-Ether 1.2
RP/0/RSP0/CPU0:Router(config-subif)# encapsulation dot1q 20
RP/0/RSP0/CPU0:Router(config-subif)# ip addr20.2.3.4/24
RP/0/RSP0/CPU0:Router(config-subifif)# no shutdown
RP/0/RSP0/CPU0:Router(config-subifif)# exit
RP/0/RSP0/CPU0:Router(config)# interface gig 0/1/5/7
RP/0/RSP0/CPU0:Router(config-if)# bundle-id 1 mode act
RP/0/RSP0/CPU0:Router(config-if)# commit
RP/0/RSP0/CPU0:Router(config-if)# exit
```

Example: Configuring a POS Link Bundle

The following example shows how to join two ports to form a Packet-over-SONET (POS) link bundle:

```
RP/0/RSP0/CPU0:Router# config
RP/0/RSP0/CPU0:Router(config)# interface Bundle-POS 5
RP/0/RSP0/CPU0:Router(config-if)# ipv4 address 1.2.3.4/24
RP/0/RSP0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/RSP0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/RSP0/CPU0:Router(config-if)# exit
RP/0/RSP0/CPU0:Router(config)# interface POS 0/0/1/1
RP/0/RSP0/CPU0:Router(config-if)# bundle id 5
RP/0/RSP0/CPU0:Router(config-if)# no shutdown
RP/0/RSP0/CPU0:Router(config-if)# exit
```

Example: Configuring EFP Load Balancing on an Ethernet Link Bundle

The following example shows how to configure all egressing traffic on the fixed members of a bundle to flow through the same physical member link automatically.

```
RP/0/RP0/CPU0:router# configuration terminal
RP/0/RP0/CPU0:router(config)# interface bundle-ether 1.1 l2transport
RP/0/RP0/CPU0:router(config-subif)# bundle load-balancing hash auto
RP/0/RP0/CPU0:router(config-subif)#
```

The following example shows how to configure all egressing traffic on the fixed members of a bundle to flow through a specified physical member link.

```
RP/0/RP0/CPU0:router# configuration terminal
RP/0/RP0/CPU0:router(config)# interface bundle-ether 1.1 l2transport
RP/0/RP0/CPU0:router(config-subif)# bundle load-balancing hash 1
RP/0/RP0/CPU0:router(config-subif)#
```

Example: Configuring Multichassis Link Aggregation

This example shows how to configure POAs:

Active POA

```
interface Bundle-Ether10
   mlacp iccp-group 1
   mlacp port-priority 10
```

Standby POA

```
interface Bundle-Ether10
   mlacp iccp-group 1
   mlacp port-priority 20
```

This example shows how to configure ICCP:

```
redundancy iccp group
   member neighbor 1.2.3.4
   backbone interface GigabitEthernet 0/0/0/0
   isolation recovery-delay 30
```

This example shows how to configure mLACP:

```
configure
 redundancy iccp group 100
   mlacp system mac 1.1.1
   mlacp system priority 10
   mlacp node 1
     interface Bundle-Ether 3
        mac-address 1.1.1
        bundle wait-while 100
        lacp switchover suppress-flaps 300
        mlacp iccp-group 100
```

This example illustrates a switchover:

```
RP/0/0/CPU0:router# show bundle

Bundle-Ether1
  Status:                                    Up
  Local links <active/standby/configured>:   1 / 0 / 1
  Local bandwidth <effective/available>:     1000000 (1000000) kbps
  MAC address (source):                    0000.deaf.0000 (Configured)
  Minimum active links / bandwidth:        1 / 1 kbps
  Maximum active links:                    64
  Wait while timer:                        100 ms
  LACP:                                    Operational
    Flap suppression timer:                300 ms
  mLACP:                                   Operational
    ICCP Group:                            1
    Role:                                  Active
    Foreign links <active/configured>:     0 / 1
    Switchover type:                       Non-revertive
    Recovery delay:                        300 s
    Maximize threshold:                    Not configured
  IPv4 BFD:                                Not configured

  Port                  Device           State        Port ID         B/W, kbps
  --------------------  ---------------  -----------  --------------  ----------
  Gi0/0/0/0             Local            Active       0x8001, 0x9001     1000000
      Link is Active
  Gi0/0/0/0             5.4.3.2          Standby      0x8002, 0xa001     1000000
      Link is marked as Standby by mLACP peer


RP/0/0/CPU0:router#mlacp switchover Bundle-Ether 1

This will trigger the peer device (Node 5.4.3.2 in IG 1) to become active for
Bundle-Ether1. This may result in packet loss on the specified bundle.

Proceed with switch over? [confirm]

RP/0/0/CPU0:Jan 31 23:46:44.666 : BM-DISTRIB[282]: %L2-BM-5-MLACP_BUNDLE_ACTIVE : This
device is no longer the active device for Bundle-Ether1
RP/0/0/CPU0:Jan 31 23:46:44.668 : BM-DISTRIB[282]: %L2-BM-6-ACTIVE :
GigabitEthernet0/0/0/0 is no longer Active as part of Bundle-Ether1 (Not enough links
available to meet minimum-active threshold)

RP/0/0/CPU0:router#show bundle
Mon Jun  7 06:04:17.778 PDT

Bundle-Ether1
  Status:                                  mLACP hot standby
  Local links <active/standby/configured>:   0 / 1 / 1
  Local bandwidth <effective/available>:     0 (0) kbps
  MAC address (source):                    0000.deaf.0000 (Configured)
  Minimum active links / bandwidth:        1 / 1 kbps
  Maximum active links:                    64
  Wait while timer:                        100 ms
  LACP:                                    Operational
    Flap suppression timer:                300 ms
  mLACP:                                   Operational
    ICCP Group:                            1
    Role:                                  Standby
    Foreign links <active/configured>:     1 / 1
    Switchover type:                       Non-revertive
    Recovery delay:                        300 s
    Maximize threshold:                    Not configured
  IPv4 BFD:                                Not configured

  Port                  Device           State        Port ID         B/W, kbps
  --------------------  ---------------  -----------  --------------  ----------
  Gi0/0/0/0             Local            Standby      0x8003, 0x9001     1000000
      mLACP peer is active
  Gi0/0/0/0             5.4.3.2          Active       0x8002, 0xa001     1000000
      Link is Active
```

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

This example shows how to add a backup pseudowire to a VPLS access pseudowire:

```
l2vpn bridge group bg1
  bridge-domain bd1
   neighbor 101.101.101.101 pw-id 5000
       pw-class class1
           backup neighbor 102.102.102.102 pw-id 3000
               pw-class class1
           !
        !
    !
  !
```

This example shows how to configure one-way pseudowire redundancy behavior when redundancy group is configured:

```
l2vpn pw-class class_mpls
   encapsulation mpls
       redundancy one-way
    !
!
```

The following example illustrates an overall MC-LAG configuration:

Topology:

```
DHD                       POA 1                      POA 2

Gi0/0/0/0 --------------- Gi0/0/0/0
Gi0/0/0/1 --------------- Gi0/0/0/1
Gi0/0/0/2
Gi0/0/0/3 ---------------------------------------- Gi0/0/0/0
Gi0/0/0/4 ---------------------------------------- Gi0/0/0/1
                          Gi0/0/0/2                 Gi0/0/0/2
                          Gi0/0/0/3 --------------- Gi0/0/0/3
                          Gi0/0/0/4 --------------- Gi0/0/0/4
```

On POA 1:

```
redundancy
 iccp
  group 1
   mlacp node 1
   mlacp system mac 000d.000e.000f
   mlacp system priority 1
   member
    neighbor 5.4.3.2
   !
  !
 !
!
interface Bundle-Ether1
 lacp switchover suppress-flaps 300
 mlacp iccp-group 1
 mac-address 0.deaf.0
 bundle wait-while 100
!
interface Loopback0
 ipv4 address 5.4.3.1 255.255.255.255
!
interface GigabitEthernet0/0/0/0
```

```
 description Connected to DHD Gi0/0/0/0
 bundle id 1 mode active
 lacp period short
 no shutdown
!
interface GigabitEthernet0/0/0/3
 description Connected to POA2 Gi0/0/0/3
 ipv4 address 1.2.3.1 255.255.255.0
 proxy-arp
 no shutdown
!
router static
 address-family ipv4 unicast
  5.4.3.2/32 1.2.3.2
 !
!
mpls ldp
 router-id 5.4.3.1
 discovery targeted-hello accept
 log
  neighbor
 !
 interface GigabitEthernet0/0/0/3
 !
!
```

On POA 2:

```
redundancy
 iccp
  group 1
   mlacp node 2
   mlacp system mac 000d.000e.000f
   mlacp system priority 1
   member
    neighbor 5.4.3.1
   !
  !
 !
!
interface Bundle-Ether1
 lacp switchover suppress-flaps 300
 mlacp iccp-group 1
 mac-address 0.deaf.0
 bundle wait-while 100
!
interface Loopback0
 ipv4 address 5.4.3.2 255.255.255.255
!
interface GigabitEthernet0/0/0/0
 description Connected to DHD Gi0/0/0/3
 bundle id 1 mode active
 lacp period short
 no shutdown
!
interface GigabitEthernet0/0/0/3
 description Connected to POA1 Gi0/0/0/3
 ipv4 address 1.2.3.2 255.255.255.0
 proxy-arp
 no shutdown
!
router static
 address-family ipv4 unicast
  5.4.3.1/32 1.2.3.1
 !
!
mpls ldp
 router-id 5.4.3.2
 discovery targeted-hello accept
 log
  neighbor
 !
 interface GigabitEthernet0/0/0/3
 !
!
```

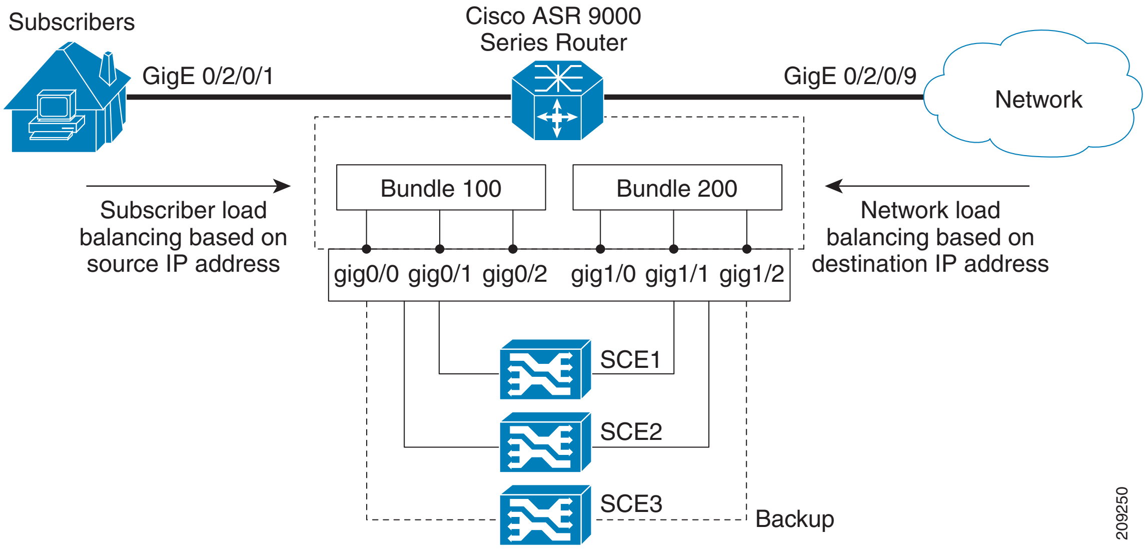
On the DHD:

```
interface Bundle-Ether1
 lacp switchover suppress-flaps 300
 bundle wait-while 100
!
interface GigabitEthernet0/0/0/0
 description Connected to POA1 Gi0/0/0/0
 bundle id 1 mode active
 lacp period short
 no shutdown
!
interface GigabitEthernet0/0/0/3
 description Connected to POA2 Gi0/0/0/0
 bundle id 1 mode active
 lacp period short
 no shutdown
!
```

Configuration Examples for MGSCP

Figure 16 illustrates a sample network with a single Cisco ASR 9000 Series Router as a dispatcher for a cluster of SCE devices that is used as an example for the sample configurations.

Figure 16 Cisco ASR 9000 Series Router as Dispatcher for SCE Cluster


Subscribers
Cisco ASR 9000 Series Router
GigE 0/2/0/1
GigE 0/2/0/9
Network
Bundle 100
Bundle 200
Subscriber load balancing based on source IP address
Network load balancing based on destination IP address
gig0/0 gig0/1 gig0/2
gig1/0 gig1/1 gig1/2
SCE1
SCE2
SCE3
Backup
209250

This section includes the following examples:

- Example: Configuring Bundle Interfaces and Member Links, page 271
- Examples: Configuring VRFs to Route Traffic to the Bundles, page 272
- Example: Configuring MGSCP with ABF to Route Traffic to the Bundles, page 274

Example: Configuring Bundle Interfaces and Member Links

The following example shows how to configure the two bundles on the Cisco ASR 9000 Series Router shown in Figure 16. Each bundle supports a maximum of two active links (configurations for both bundles must match), with one backup protect link.

The bundle interface members in Ethernet bundle 100 connect the SCE device links for the subscriber side of the network using load balancing based on source IP address. The bundle interface members in Ethernet bundle 200 connect the SCE device links for the core side of the network using load balancing based on destination IP address.

Subscriber-Facing Access Bundle Configuration

```
interface Bundle-Ether 100
 description Subscriber-facing end
 vrf access
 ipv4 address 10.10.1.2 255.255.255.0
 lacp cisco enable link-order signaled
 bundle load-balancing hash src-ip
 bundle maximum-active links 2
!
interface GigabitEthernet 0/0/0/0
 description to SCE1
 bundle id 100 mode active
 bundle port-priority 1
!
interface GigabitEthernet 0/0/0/1
 description to SCE2
 bundle id 100 mode active
 bundle port-priority 1
!
interface GigabitEthernet 0/0/0/3
 description to SCE3 (backup)
 bundle id 100 mode active
```

Core-Facing Bundle Configuration

```
interface Bundle-Ether 200
 description Core-facing end
 vrf core
 ipv4 address 10.20.1.2 255.255.255.0
 lacp cisco enable link-order signaled
 bundle load-balancing hash dst-ip
 bundle maximum active links 2
!
interface GigabitEthernet 0/0/1/0
 description from SCE1
 bundle id 200 mode active
 bundle port-priority 1
!
interface GigabitEthernet 0/0/1/1
 description from SCE2
 bundle id 200 mode active
 bundle port-priority 1
```

```
!
interface GigabitEthernet 0/0/1/2
 description from SCE3 (standby)
 bundle id 200 mode active
```

Examples: Configuring VRFs to Route Traffic to the Bundles

To ensure that the traffic to and from the same subscriber is going through the same port of the SCE, VRFs are recommended. You need to configure two VRFs for MGSCP: One for the access traffic, and one for the core traffic.

The examples in this section also show two different ways that you can route using VRFs with either static or dynamic (OSPF) routing for the bundle interface at the VRF:

- Example: Configuring VRFs with Static Routing, page 272
- Example: Configuring VRFs with OSPF Routing, page 273

Example: Configuring VRFs with Static Routing

In the following configuration examples, VRFs are established for the core and access sides of the network using IPv4. From there, the bundle interface addresses for each side are each configured as part of the VRF, as well as two physical interfaces. The final piece of the configuration shows how to configure a static route to each VRF using the bundle interfaces.

VRF Global Configuration

```
vrf core
 address-family ipv4 unicast
  import route-target
   1:1
  !
  export route-target
   1:1
  !
vrf access
 address-family ipv4 unicast
  import route-target
   1:1
  !
  export route-target
   1:1
  !
```

VRF Configuration on Bundle Interfaces

```
interface Bundle-Ether100
 vrf access
 ipv4 address 10.10.1.2 255.255.255.0
!
interface Bundle-Ether200
 vrf core
 ipv4 address 10.20.1.2 255.255.255.0
```

VRF Configuration on Physical Interfaces

```
interface GigabitEthernet0/2/0/1
 vrf access
 ipv4 address 10.10.1.4 255.255.255.0
```

```
interface GigabitEthernet0/2/0/9
 vrf core
 ipv4 address 10.20.1.4 255.255.255.0
 negotiation auto
```

Static Routing Configuration for the VRFs to the Bundle Interfaces

```
router static
 vrf core
  address-family ipv4 unicast
   0.0.0.0/0 Bundle-Ether200
  !
 !
 vrf access
  address-family ipv4 unicast
   0.0.0.0/0 Bundle-Ether100
  !
 !
```

Example: Configuring VRFs with OSPF Routing

In the following configuration examples, VRFs are established for the core and access sides of the network using IPv4. From there, you configure an OSPF routing instance and area to include the VRFs and associate the bundle and physical interfaces.

Global VRF Configuration

```
vrf core
 address-family ipv4 unicast
  import route-target
   1:1
  export route-target
   1:1

vrf access
 address-family ipv4 unicast
  import route-target
   1:1
  export route-target
   1:1
```

VRF Configuration on Physical Interfaces

```
interface GigabitEthernet0/2/0/1
 vrf access
 ipv4 address 10.10.1.4 255.255.255.0
_
interface GigabitEthernet0/2/0/9
 vrf core
 ipv4 address 10.20.1.4 255.255.255.0
```

OSPF Routing Configuration for the VRFs and the Bundle and Physical Interfaces

```
router ospf 100
 vrf core
  router-id 10.20.1.2
  area 0
   interface Bundle-Ether200
   interface GigabitEthernet0/2/0/9

 vrf access
  router-id 10.10.1.2
  area 0
```

```
   interface Bundle-Ether100
   interface GigabitEthernet0/2/0/1
```

Example: Configuring MGSCP with ABF to Route Traffic to the Bundles

The following example routes traffic to the bundles using access lists to forward the traffic.

```
ipv4 access-list inbound
!
! Set the nexthop address to be a virtual IP address on the same network
! as the access bundle.
!
 10 permit ipv4 any any nexthop 10.10.1.5
!
ipv4 access-list outbound
!
! Set the nexthop address to be a virtual IP address on the same network
! as the core bundle.
!
 10 permit ipv4 any any nexthop 10.20.1.5
!
! Configure static ARP for the virtual IP addresses
!
arp vrf default 10.10.1.5 0024.98eb.bf8a ARPA
arp vrf default 10.20.1.5 0024.98eb.bf8b ARPA

interface Bundle-Ether100
 ipv4 address 10.10.1.2 255.255.255.0
!
interface Bundle-Ether200
 ipv4 address 10.20.1.2 255.255.255.0
!
interface GigabitEthernet0/2/0/1
 ipv4 address 10.10.1.3 255.255.255.0
 ipv4 access-group inbound
!
interface GigabitEthernet0/2/0/9
ipv4 address 10.20.1.3 255.255.255.0
 ipv4 access-group outbound
!
```

Additional References

The following sections provide references related to link bundle configuration.

Related Documents

Related Topic	Document Title
Cisco ASR 9000 Series Router master command reference	*Cisco ASR 9000 Series Router Master Commands List*
Cisco ASR 9000 Series Router interface configuration commands	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a Cisco ASR 9000 Series Router using the Cisco IOS XR Software.	*Cisco ASR 9000 Series Router Getting Started Guide*
Information about user groups and task IDs	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Traffic Mirroring on the Cisco ASR 9000 Series Router

This module describes the configuration of traffic mirroring on the Cisco ASR 9000 Series Router. Traffic mirroring is sometimes called port mirroring, or switched port analyzer (SPAN).

Feature History for Configuring Traffic Mirroring on the Cisco ASR 9000 Series Router

Release	Modification
Release 3.9.1	This feature was introduced on the Cisco ASR 9000 Series Router.
Release 4.0.1	The following traffic mirroring features were added: • Traffic mirroring over a pseudowire • Flow or ACL-based traffic mirroring • Layer 3 interface support • Partial packet mirroring

Contents

Restrictions for Traffic Mirroring

A maximum of eight monitoring sessions, and 800 source ports are supported.

You can configure 800 source ports on a single monitoring session, or configure an aggregate total of 800 source ports on a maximum of eight monitoring sessions.

The following forms of traffic mirroring are not supported:

- Mirroring traffic to a GRE tunnel (also known as Encapsulated Remote Switched Port Analyzer [ER-SPAN] in Cisco IOS Software).
- Mirroring traffic from a full bridge domain (also known as VLAN-based SPAN in Cisco IOS Software).
- Mirroring traffic on an individual bundle member interface is not supported. SPAN must only be configured on a bundle interface and it is applied to all members.

Performance Impact with Traffic Mirroring

It is recommended that you do not mirror more than 15% of your total transit traffic. On the Cisco ASR 9000 Ethernet Line Card, that uses Ten Gigabit Ethernet interfaces or bundle interfaces there is a limit of 1.5G of data on each of the ingress and egress traffic that can be mirrored. This limitation is not applicable on the Cisco ASR 9000 Enhanced Ethernet Line Card.

Information about Traffic Mirroring

The following sections provide information about traffic mirroring:

Introduction to Traffic Mirroring

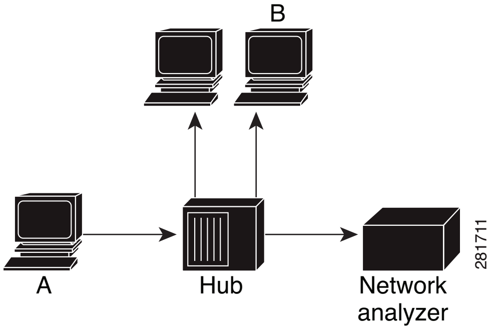
Traffic mirroring, which is sometimes called port mirroring, or Switched Port Analyzer (SPAN) is a Cisco proprietary feature that enables you to monitor Layer 2 or Layer 3 network traffic passing in, or out of, a set of Ethernet interfaces. You can then pass this traffic to a network analyzer for analysis.

Traffic mirroring copies traffic from one or more Layer 3 interfaces or Layer 2 interfaces or sub-interfaces, including Layer 2 link bundle interfaces or sub-interfaces, and sends the copied traffic to one or more destinations for analysis by a network analyzer or other monitoring device. Traffic mirroring does not affect the switching of traffic on the source interfaces or sub-interfaces, and allows the mirrored traffic to be sent to a destination interface or sub-interface.

Traffic mirroring was introduced on switches because of a fundamental difference between switches and hubs. When a hub receives a packet on one port, the hub sends out a copy of that packet from all ports except from the one to which the hub received the packet. In the case of switches, after a switch boots, it starts to build up a Layer 2 forwarding table on the basis of the source MAC address of the different packets that the switch receives. After this forwarding table is built, the switch forwards traffic that is destined for a MAC address directly to the corresponding port.

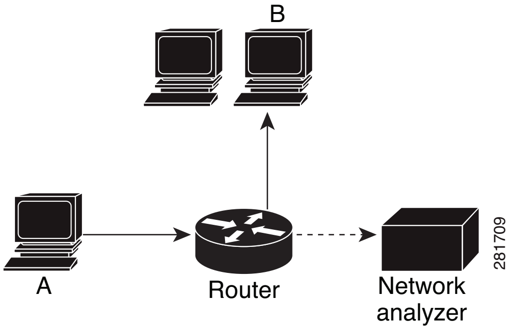
For example, if you want to capture Ethernet traffic that is sent by host A to host B, and both are connected to a hub, just attach a traffic analyzer to this hub. All other ports see the traffic between hosts A and B (Figure 17).

Figure 17 Traffic Mirroring Operation on a Hub



On a switch or router, after the host B MAC address is learned, unicast traffic from A to B is only forwarded to the B port. Therefore, the traffic analyzer does not see this traffic (Figure 18).

Figure 18 Network Analysis Does Not Work on a Router Without Traffic Mirroring



In this configuration, the traffic analyzer only captures traffic that is flooded to all ports, such as:

- Broadcast traffic
- Multicast traffic with CGMP or Internet Group Management Protocol (IGMP) snooping disabled
- Unknown unicast traffic on a switch

An extra feature is necessary that artificially copies unicast packets that host A sends. This extra feature is traffic mirroring. When traffic mirroring is enabled, the traffic analyzer is attached to a port that is configured to receive a copy of every packet that host A sends. This port is called a traffic mirroring port. The other sections of this document describe how you can fine tune this feature.

Implementing Traffic Mirroring on the Cisco ASR 9000 Series Router

Traffic Mirroring Terminology

- Ingress traffic—Traffic that enters the switch.
- Egress traffic—Traffic that leaves the switch.

- Source port—A port that is monitored with the use of traffic mirroring. It is also called a monitored port.
- Destination port—A port that monitors source ports, usually where a network analyzer is connected. It is also called a monitoring port.
- Monitor session—A designation for a collection of traffic mirroring configurations consisting of a single destination and, potentially, many source interfaces.

Characteristics of the Source Port

A source port, also called a monitored port, is a switched or routed port that you monitor for network traffic analysis. In a single local or remote traffic mirroring session, you can monitor source port traffic, such as received (Rx) for ingress traffic, transmitted (Tx) for egress traffic, or bidirectional (for both ingress and egress traffic). Your router supports any number of source ports (up to the maximum number of 800).

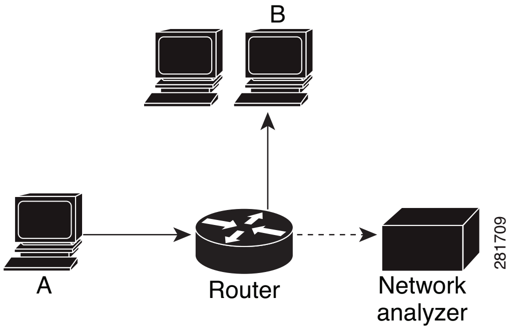
A source port has these characteristics:

- It can be any port type, such as Bundle Interface, Gigabit Ethernet, 10-Gigabit Ethernet, or EFPs.

Note Bridge group virtual interfaces (BVIs) are not supported.

- Each source port can be monitored in one traffic mirroring session.
- It cannot be a destination port.
- Each source port can be configured with a direction (ingress, egress, or both) to monitor. For bundles, the monitored direction applies to all physical ports in the group.

Figure 19 Network Analysis on a Cisco ASR 9000 Router With Traffic Mirroring



In Figure 19, the network analyzer is attached to a port that is configured to receive a copy of every packet that host A sends. This port is called a traffic mirroring port.

Characteristics of the Monitor Session

A monitor session is a collection of traffic mirroring configurations consisting of a single destination and, potentially, many source interfaces. For any given monitor session, the traffic from the source interfaces (called *source ports*) is sent to the monitoring port (called the *destination port*). Some optional operations such as VLAN tag imposition and ACL filtering can be performed on the mirrored traffic streams. If there is more than one source port in a monitoring session, the traffic from the several mirrored traffic streams is combined at the destination port. The result is that the traffic that comes out of the destination port is a combination of the traffic from one or more source ports, and the traffic from each source port may or may not have VLAN push operations or ACLs applied to it.

Monitor sessions have the following characteristics:

- A single Cisco ASR 9000 Router can have a maximum of eight monitor sessions.
- A single monitor session can have only one destination port.
- A single destination port can belong to only one monitor session.
- A single Cisco ASR 9000 Router can have a maximum of 800 source ports.
- A monitor session can have a maximum of 800 source ports, as long as the maximum number of source ports from all monitoring sessions does not exceed 800.

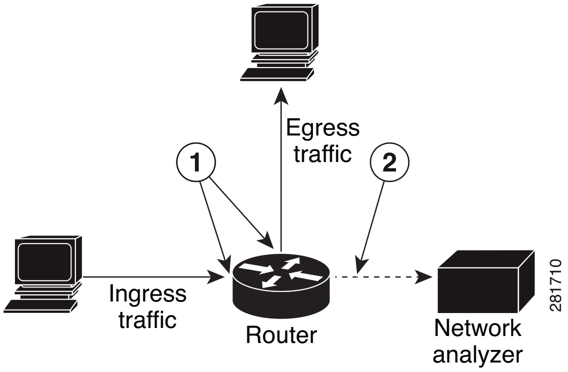
Characteristics of the Destination Port

Each local session or remote destination session must have a destination port (also called a monitoring port) that receives a copy of the traffic from the source ports.

A destination port has these characteristics:

- A destination port must reside on the same router as the source port.
- A destination port can be any Ethernet physical port, EFP, pseudowire, but not a bundle interface.
- A destination port can only be a Layer 2 transport interface. A Layer 3 interface as a SPAN destination is not supported on the Cisco ASR 9000 Series Router.
- A destination port and can be a trunk (main) interface or a subinterface.
- At any one time, a destination port can participate in only one traffic mirroring session. A destination port in one traffic mirroring session cannot be a destination port for a second traffic mirroring session. In other words, no two monitor sessions can have the same destination port.
- A destination port cannot also be a source port.

Figure 20 Network Analysis on a Cisco ASR 9000 Router With Traffic Mirroring



1	Source traffic mirroring ports (can be ingress or egress traffic ports)	**2**	Destination traffic mirroring port

Supported Traffic Mirroring Types

The following traffic mirroring types are supported:

- Local traffic mirroring. This is the most basic form of traffic mirroring. The network analyzer or sniffer is directly attached to the destination interface. In other words, all monitored ports are all located on the same switch as the destination port.
- Remote traffic mirroring (known as R-SPAN). In this case, the network analyzer is not attached directly to the destination interface, but is on a VLAN accessible to the switch. For example, the destination interface is a sub-interface with a VLAN encapsulation.

  A restricted form of remote traffic mirroring can be implemented by sending traffic to a single destination port that pushes a VLAN tag, instead of switching via a bridge domain.

  - Allows decoupling the network analyzer and destination, but there is no on-the-box redundancy.
  - Allows multiple remote network analyzers as long as they can attach to the traffic mirroring VLAN.

  This is supported on Cisco IOS XR software, because the destination port is an EFP that can push a VLAN tag.
- Pseudowire traffic mirroring (known as PW-SPAN in Cisco IOS Software). Instead of using a standard destination interface, traffic is mirrored to a remote site via an MPLS pseudowire.
- ACL-based traffic mirroring. Traffic is mirrored based on the configuration of the global interface ACL.
- Partial Packet Mirroring. The first 64 to 256 bytes of the packet can be mirrored.
- Layer 2 or Layer 3 traffic mirroring is supported. Both Layer 2 and Layer 3 source ports can be mirrored.

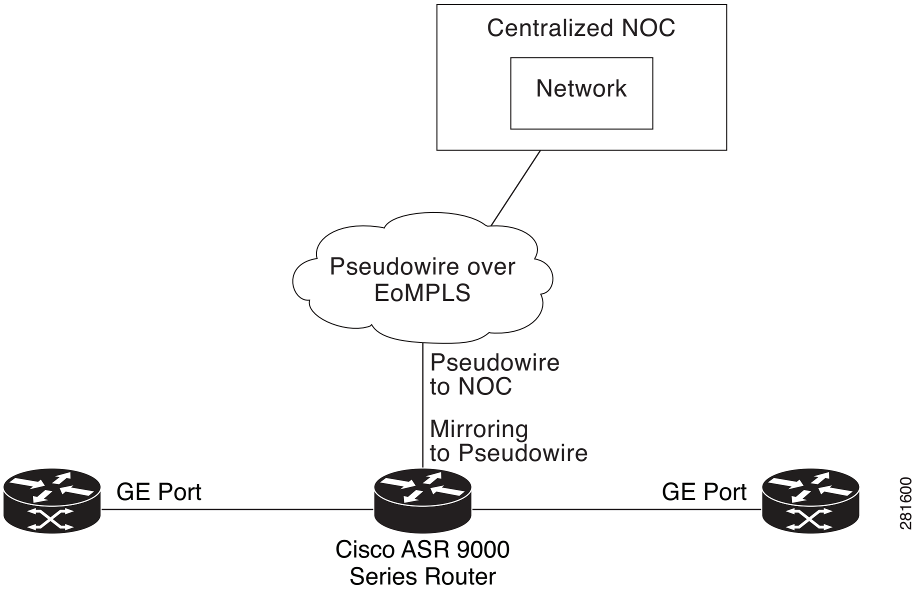
Pseudowire Traffic Mirroring

The traffic mirroring destination port can be configured to be a pseudowire rather than a physical port. In this case, the designated traffic on the source port is mirrored over the pseudowire to a central location. This allows the centralization of expensive network traffic analysis tools.

Because the pseudowire is carrying only the mirrored traffic, this traffic is generally unidirectional. There should not be any traffic coming from the remote provider edge.

To protect the pseudowire traffic mirroring path against network failures, it is possible to configure a traffic engineering tunnel as the preferred path and enable fast reroute protection for the pseudowire.

Figure 21 Pseudowire Traffic Mirroring



ACL-Based Traffic Mirroring

You can mirror traffic based on the definition of a global interface access list (ACL). If you are mirroring Layer 2 traffic, the ACL is configured using the **ethernet-services access-list** command with the **capture** keyword. If you are mirroring Layer 3 traffic, the ACL is configured using the **ipv4 access-list** or **ipv6 access-list** command with the **capture** keyword. The **permit** and **deny** commands determine the behavior of regular traffic. The **capture** keyword designates that the packet is to be mirrored to the destination port.

Configuring Traffic Mirroring

The following tasks describe how to configure traffic mirroring:

- How to Configure Local Traffic Mirroring, page 283
- How to Configure Remote Traffic Mirroring, page 285
- How to Configure Traffic Mirroring over Pseudowire, page 287
- How to Configure ACL-Based Traffic Mirroring, page 291
- How to Configure Partial Packet Mirroring, page 294

How to Configure Local Traffic Mirroring

SUMMARY STEPS

1. **configure**
2. **monitor-session** *session-name*
3. **destination interface** *dest-interface*

4. **exit**
5. **interface** *source-interface*
6. **l2transport**
7. **monitor-session** *session-name* [**direction** {**rx-only** | **tx-only**]
8. **end**
   or
   commit
9. **show monitor-session** [*session-name*] **status** [**detail**] [**error**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**monitor-session** *session-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# monitor-session mon1`
	`RP/0/RSP0/CPU0:router(config-mon)#`
		Defines a monitor session, and enters monitor session configuration mode.

Step 3	**destination interface** *dest-interface* **Example:** `RP/0/RSP0/CPU0:router(config-mon)# destination interface gigabitethernet0/0/0/15`	Specifies the destination interface to which traffic should be replicated. This interface must be a Layer 2 transport interface.

Step 4	**exit**
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# exit`
	`RP/0/RSP0/CPU0:router(config)#`
		Exits monitor session configuration mode, and returns to global configuration mode.

Step 5	**interface** *source-interface* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet0/0/0/11`	Enters interface configuration mode for the specified interface. The interface number is entered in *rack/slot/module/port* notation. For more information about the syntax for the router, use the question mark (?) online help function.
Step 6	**l2transport** **Example:** `RP/0/RSP0/CPU0:router(config-if)# l2transport`	(Optional) Enables Layer 2 transport mode on the interface and enters Layer 2 transport configuration mode. **Note** Use the **l2transport** command to mirror all traffic types.

	Command or Action	Purpose
Step 7	**monitor-session** *session-name* [**direction** {**rx-only** | **tx-only**] **Example:** `RP/0/RSP0/CPU0:router(config-if-l2)# monitor-session mon1`	Specifies the monitor session to be used on this interface. Use the **direction** keyword to specify that only ingress or only egress traffic is mirrored.

Step 8	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if-l2)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 9	**show monitor-session [session-name**] **status** [**detail**] [**error**] **Example:** `RP/0/RSP0/CPU0:router# show monitor-session`	Displays information about the monitor session.

How to Configure Remote Traffic Mirroring

SUMMARY STEPS

1. **configure**
2. **monitor-session** *session-name*
3. **destination interface** *dest-subinterface*
4. **exit**
5. **interface** *dest-subinterface* **l2transport**
6. **encapsulation dot1q** *vlan*
7. **rewrite ingress tag pop** *tag-to-remove*
8. **interface** *source-interface* [**l2transport**]
9. **monitor-session** *session-name* [**direction** {**rx-only** | **tx-only**]

10. **end**
    or
    commit
11. **show monitor-session** [*session-name*] **status** [**detail**] [**error**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`monitor-session`** *`session-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# monitor-session mon1`
	`RP/0/RSP0/CPU0:router(config-mon)#`
		Defines a monitor session, and enters monitor session configuration mode.

Step 3	**`destination interface`** *`dest-subinterface`*
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# destination interface gigabitethernet0/0/0/15`
		Specifies the destination subinterface to which traffic should be replicated. This interface must be a Layer 2 transport interface.

Step 4	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# exit`
	`RP/0/RSP0/CPU0:router(config)#`
		Exits monitor session configuration mode, and returns to global configuration mode.

Step 5	**`interface`** *`dest-subinterface`* **`l2transport`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface gigabitethernet0/0/0/11.10 l2transport`
		Enters interface configuration mode for the specified sub-interface. The interface number is entered in *rack/slot/module/port* notation. For more information about the syntax for the router, use the question mark (?) online help function.
		The **l2transport** keyword is used to enable Layer 2 transport mode on the destination subinterface.

Step 6	**`encapsulation dot1q`** *`vlan`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1q 1`
		Specifies 802.1Q encapsulation and the VLAN number that is used.

Step 7	**`rewrite ingress tag pop`** *`tag-to-remove`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# rewrite ingress tag pop 1`
		Specifies to remove the outer tag only for the EFP.

	Command or Action	Purpose
Step 8	**interface** *source-subinterface* [**l2transport**] **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet0/0/0/11.10 l2transport`	Enters interface configuration mode for the specified subinterface. The interface number is entered in *rack/slot/module/port* notation. For more information about the syntax for the router, use the question mark (?) online help function. To configure a Layer 2 subinterface to be the source interface, use the **l2transport** keyword to enable Layer 2 transport mode on the subinterface.
Step 9	**monitor-session** *session-name* [**direction** {**rx-only** | **tx-only**] **Example:** `RP/0/RSP0/CPU0:router(config-if-l2)# monitor-session mon1`	Specifies the monitor session to be used on this interface. Use the **direction** keyword to specify that only ingress or egress traffic is mirrored.

Step 10	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 11	**show monitor-session [session-name]** **status** [**detail**] [**error**] **Example:** `RP/0/RSP0/CPU0:router# show monitor-session`	Displays information about the traffic mirroring session.

How to Configure Traffic Mirroring over Pseudowire

SUMMARY STEPS

1. **configure**
2. **monitor-session** *session-name*
3. **destination pseudowire**

4. **exit**
5. **interface** *source-interface*
6. **l2transport**
7. **monitor-session** *session-name*
8. **exit**
9. **exit**
10. **exit**
11. **l2vpn**
12. **pw-class** *class-name*
13. **encapsulation mpls**
14. **exit**
15. **exit**
16. **xconnect group** *group-name*
17. **p2p** *xconnect-name*
18. **monitor-session** *session-name*
19. **neighbor** *peer-ip* **pw-id** *pseudowire-id*
20. **pw-class** *class-name*
21. **end**
    or
    commit
22. **show monitor-session** [*session-name*] **status** [**detail**] [**error**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`monitor-session`** *`session-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# monitor-session mon1`
	`RP/0/RSP0/CPU0:router(config-mon)#`
		Defines a monitor session, and enters monitor session configuration mode.

Step 3	**`destination psuedowire`**
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# destination pseudowire`
		Specifies that the traffic should be replicated to a pseudowire.

	Command or Action	Purpose

Step 4	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# exit`
	`RP/0/RSP0/CPU0:router(config)#`
		Exits monitor session configuration mode and returns to global configuration mode.

Step 5	**`interface`** *`source-interface`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface gigabitethernet0/0/0/11.10`
		Enters interface configuration mode for the specified interface. The interface number is entered in *rack/slot/module/port* notation. For more information about the syntax for the router, use the question mark (?) online help function.

Step 6	**`l2transport`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# l2transport`
		(Optional) Enables Layer 2 transport mode on the subinterface and enters Layer 2 transport configuration mode.
		Note Use the **l2transport** command to mirror all traffic types.

Step 7	**`monitor-session`** *`session-name`* `[`**`direction`** `{`**`rx-only`** `|` **`tx-only`**`]`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# monitor-session mon1`
		Specifies the monitor session to be used on this interface. Use the **direction** keyword to specify that only ingress or egress traffic is mirrored.

Step 8	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-mon)# exit`
	`RP/0/RSP0/CPU0:router(config-if-l2)#`
		Exits monitor session configuration mode and returns to l2transport configuration mode.

Step 9	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# exit`
	`RP/0/RSP0/CPU0:router(config-if)#`
		Exits l2transport configuration mode and returns to interface configuration mode.

Step 10	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
	`RP/0/RSP0/CPU0:router(config)#`
		Exits interface configuration mode and returns to global configuration mode.

Step 11	**`l2vpn`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# l2vpn`
	`RP/0/RSP0/CPU0:router(config-l2vpn)#`
		Enters Layer 2 VPN configuration mode.

Step 12	**`pw-class`** *`class-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn)# pw-class pw1`
		Configures a pseudowire class template and enters pseudowire class template configuration mode.

	Command or Action	Purpose
Step 13	**encapsulation mpls** **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls`	Configures the pseudowire encapsulation to MPLS.

Step 14	**exit**
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mpls)# exit`
	`RP/0/RSP0/CPU0:router(config-l2vpn-pwc)`
		Exits pseudowire encapsulation configuration mode.

Step 15	**exit**
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# exit`
	`RP/0/RSP0/CPU0:router(config-l2vpn)`
		Exits pseudowire class template configuration mode.

Step 16	**xconnect group** *group-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group g1`	Configures a group cross connect.
Step 17	**p2p** *xconnect-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p xc1`	Configures a point-to-point cross connect.
Step 18	**monitor-session** *session-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# monitor-session mon1`	Attaches a traffic mirroring session to the point-to-point cross connect.
Step 19	**neighbor** *peer-ip* **pw-id** *pseudowire-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor 192.168.2.2 pw-id 3`	Configures the point-to-point cross connect.
Step 20	**pw-class** *class-name* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# pw-class pw1`	Specifies the pseudowire class template to use for this cross connect.

	Command or Action	Purpose

Step 21	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# end`
	or
	`RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 22	**show monitor-session [session-name] status** [**detail**] [**error**] **Example:** `RP/0/RSP0/CPU0:router# show monitor-session`	Displays information about the traffic mirroring session.

How to Configure ACL-Based Traffic Mirroring

Prerequisites

The global interface ACL should be configured using one of the following commands with the **capture** keyword:

- **ipv4 access-list**
- **ipv6 access-list**
- **ethernet-services access-list**

For more information, refer to the *Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Command Reference* or the *ASR 9000 Series Aggregation Services Router L2 VPN and Ethernet Services Command Reference*.

SUMMARY STEPS

1. **configure**
2. **monitor-session** *session-name*
3. **destination interface** *dest-interface*
4. **exit**

5. **interface** *source-interface*
6. **l2transport**
7. **exit**
8. **ethernet-services access-group** *access-list-name* **ingress**
9. **acl**
10. **monitor-session** *session-name*
11. **end**
    or
    commit
12. **show monitor-session** [*session-name*] **status** [**detail**] [**error**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**monitor-session** *session-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# monitor-session mon1`
	`RP/0/RSP0/CPU0:router(config-mon)#`
		Defines a monitor session and enters monitor session configuration mode.

Step 3	**destination interface** *dest-interface* **Example:** `RP/0/RSP0/CPU0:router(config-mon)# destination interface gigabitethernet0/0/0/15`	Specifies the destination interface to which traffic should be replicated. This interface must be a Layer 2 transport interface.

Step 4	**exit**
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# exit`
	`RP/0/RSP0/CPU0:router(config)#`
		Exits monitor session configuration mode and returns to global configuration mode.

Step 5	**interface** *source-interface* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet0/0/0/11`	Enters interface configuration mode for the specified interface. The interface number is entered in *rack*/*slot*/*module*/*port* notation. For more information about the syntax for the router, use the question mark (?) online help function.
Step 6	**l2transport** **Example:** `RP/0/RSP0/CPU0:router(config-if)# l2transport`	(Optional) Enables Layer 2 transport mode on the subinterface and enters Layer 2 transport configuration mode. **Note** Use the **l2transport** command to mirror all traffic types.

	Command or Action	Purpose

Step 7	**exit**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# exit`
	`RP/0/RSP0/CPU0:router(config-if)#`
		Exits Layer 2 transport configuration mode and returns to interface configuration mode.

Step 8	**ethernet-services access-group** *access-list-name* [**ingress** | **egress**]
	Example:
	`RP/0/RSP0/CPU0:router(config-if)#`
	`ethernet-services access-group acl1 ingress`
		Associates the access list definition with the interface being mirrored.

Step 9	**acl** **Example:** `RP/0/RSP0/CPU0:router(config-if-mon)# acl`	Specifies that the traffic mirrored is according to the defined global interface ACL.

Step 10	**monitor-session** *session-name*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)#`
	`monitor-session mon1`
		Specifies the monitor session to be used on this interface.

Step 11	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 12	**show monitor-session [session-name]** **status** [**detail**] [**error**] **Example:** `RP/0/RSP0/CPU0:router# show monitor-session`	Displays information about the monitor session.

Troubleshooting ACL-Based Traffic Mirroring

Note the following configuration issues:

- Even when the **acl** command is configured on the source mirroring port, if the ACL configuration command does not use the **capture** keyword, no traffic gets mirrored.
- If the ACL configuration uses the **capture** keyword, but the **acl** command is not configured on the source port, although traffic is mirrored, no access list configuration is applied.
- All ingress traffic is mirrored, regardless of the ACL definition; only egress traffic permitted in the ACL definition is mirrored.

The following example correctly shows both the **capture** keyword in the ACL definition and the **acl** command configured on the interface:

```
monitor-session tm_example
!
ethernet-services access-list tm_filter
 10 deny 0000.1234.5678 0000.abcd.abcd any capture
!
interface GigabitEthernet0/2/0/0
 monitor-session tm_example direction rx-only
  acl
 !
 l2transport
 !
 ethernet-services access-group tm_filter ingress
!
end
```

How to Configure Partial Packet Mirroring

SUMMARY STEPS

1. **configure**
2. **monitor-session** *session-name*
3. **destination interface** *dest-interface*
4. **exit**
5. **interface** *source-interface*
6. **monitor-session** *session-name*
7. **mirror first** *bytes*
8. **end**
   or
   commit
9. **show monitor-session** [*session-name*] **status**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`monitor-session`** *`session-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# monitor-session mon1`
	`RP/0/RSP0/CPU0:router(config-mon)#`
		Defines a monitor session, and enters monitor session configuration mode.

Step 3	**`destination interface`** *`dest-interface`*
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# destination interface gigabitethernet0/0/0/15`
		Specifies the destination interface to which traffic should be replicated. This interface must be a Layer 2 transport interface.

Step 4	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-mon)# exit`
	`RP/0/RSP0/CPU0:router(config)#`
		Exits monitor session configuration mode and returns to global configuration mode.

Step 5	**`interface`** *`source-interface`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface gigabitethernet0/0/0/11.10`
		Enters interface configuration mode for the specified interface. The interface number is entered in *rack/slot/module/port* notation. For more information about the syntax for the router, use the question mark (?) online help function.

Step 6	**`monitor-session`** *`session-name`* [**`direction`** {**`rx-only`** | **`tx-only`**] **Example:** `RP/0/RSP0/CPU0:router(config-if-l2)# monitor-session mon1`	Specifies the monitor session to be used on this interface. Use the **direction** keyword to specify that only ingress or egress traffic is mirrored.

Step 7	**`mirror first`** *`bytes`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-mon)# mirror first bytes`
		Specifies the number of bytes of the packet to mirror. Values can range from 64 to 256 bytes.

	Command or Action	Purpose

Step 8	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 9	**show monitor-session [*session-name*] status** **Example:** `RP/0/RSP0/CPU0:router# show monitor-session`	Displays information about the traffic mirroring session.

Traffic Mirroring Configuration Examples

This section contains examples of how to configure traffic mirroring:

- Traffic Mirroring with Physical Interfaces (Local): Example, page 296
- Traffic Mirroring with EFPs (Remote): Example, page 297
- Viewing Monitor Session Status: Example, page 297
- Monitor Session Statistics: Example, page 298
- Traffic Mirroring over Pseudowire: Example, page 299
- Layer 3 ACL-Based Traffic Mirroring: Example, page 299
- Layer 2 ACL-Based Traffic Mirroring: Example, page 299

Traffic Mirroring with Physical Interfaces (Local): Example

The following example shows a basic configuration for traffic mirroring with physical interfaces. When traffic flows over the point to point cross connect between gig0/2/0/19 and gig0/2/0/11, packets received and transmitted on gig0/2/0/19 are also mirrored to gig0/2/0/15.

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# monitor-session ms1
RP/0/RSP0/CPU0:router(config-mon)# destination interface gig0/2/0/15
RP/0/RSP0/CPU0:router(config-mon)# commit
```

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/11
RP/0/RSP0/CPU0:router(config-subif)# l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/15
RP/0/RSP0/CPU0:router(config-subif)# l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/19
RP/0/RSP0/CPU0:router(config-subif)# l2transport
RP/0/RSP0/CPU0:router(config-subif-l2)# monitor-session ms1
RP/0/RSP0/CPU0:router(config-if-l2)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group xg1
RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p xg1_p1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface gig0/2/0/11
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface gig0/2/0/19
RP/0/RSP0/CPU0:router(config-if-l2)# commit
```

Traffic Mirroring with EFPs (Remote): Example

The following example shows a basic configuration for remote traffic mirroring with EFP interfaces. When traffic flows over the point-to-point cross connect between gig0/2/0/19.10 and gig0/2/0/11.10, packets received and transmitted on gig0/2/0/19.10 are also mirrored to gig0/2/0/10.1.

```
RP/0/RSP0/CPU0:router#monitor-session ms1
RP/0/RSP0/CPU0:router(config)# destination interface gig0/2/0/10.1

RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/10.1 l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# encapsulation dot1q 1
RP/0/RSP0/CPU0:router(config-if-l2)# rewrite ingress tag pop 1

RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/11.10 l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# encapsulation dot1q 10

RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/19.10 l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# encapsulation dot1q 10
RP/0/RSP0/CPU0:router(config-if-l2)# monitor-session ms1

RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group xg1
RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p xg1_p1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface gig0/2/0/11.10
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface gig0/2/0/19.10
```

Viewing Monitor Session Status: Example

The following examples show sample output of the **show monitor-session** command with the **status** keyword:

```
RP/0/RSP0/CPU0:router# show monitor-session status

Fri Feb 20 14:56:04.233 UTC
```

```
Monitor-session cisco-rtp1
Destination interface GigabitEthernet0/5/0/38
================================================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------
Gi0/5/0/4             Both  Operational
Gi0/5/0/17            Both  Operational

RP/0/RSP0/CPU0:router# show monitor-session status detail

Monitor-session sess1
  Destination interface is not configured
  Source Interfaces
  -----------------
  GigabitEthernet0/0/0/0
    Direction: Both
    ACL match: Enabled
    Portion:   Full packet
    Status:    Not operational (destination interface not known).
  GigabitEthernet0/0/0/2
    Direction: Both
    ACL match: Disabled
    Portion:   First 100 bytes
    Status:    Error: 'Viking SPAN PD' detected the 'warning' condition 'PRM connection 
creation failure'.

RP/0/RSP0/CPU0:router# show monitor-session status error

Thu Jul  1 17:56:24.190 DST
Monitor-session ms1
Destination interface GigabitEthernet0/2/0/15 is not configured
================================================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------

Monitor-session ms2
Destination interface is not configured
================================================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------
RP/0/RSP0/CPU0:router#
```

Monitor Session Statistics: Example

Use the **show monitor-session** command with the **counters** keyword to show the statistics/counters (received/transmitted/dropped) of different source ports. For each monitor session, this command displays a list of all source interfaces and the replicated packet statistics for that interface.

The full set of statistics displayed for each interface is:

- RX replicated packets and octets
- TX replicated packets and octets
- Non-replicated packet and octets

```
RP/0/RSP0/CPU0:router# show monitor-session counters

Monitor-session ms1
  GigabitEthernet0/2/0/19.10
    Rx replicated: 1000 packets, 68000 octets
    Tx replicated: 1000 packets, 68000 octets
    Non-replicated: 0 packets, 0 octets
```

Use the **clear monitor-session counters** command to clear any collected statistics. By default this command clears all stored statistics; however, an optional interface filter can be supplied.

```
RP/0/RSP0/CPU0:router# clear monitor-session counters
```

Traffic Mirroring over Pseudowire: Example

The following example shows how to configure traffic mirroring over a pseudowire:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet0/11/0/1
RP/0/RSP0/CPU0:router(config-if)# l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# monitor-session pw-span-test

RP/0/RSP0/CPU0:router(config)# monitor-session pw-span-test
RP/0/RSP0/CPU0:router(config-mon)# destination pseudowire

RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# pw-class class1
RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls

RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group g1
RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p x1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# monitor-session pw-span-test
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor 2.2.2.2 pw-id 1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# pw-class class1

RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# commit
```

Layer 3 ACL-Based Traffic Mirroring: Example

The following example shows how to configure Layer 3 ACL-based traffic mirroring:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# monitor-session ms1
RP/0/RSP0/CPU0:router(config-mon)# destination interface gig0/2/0/15
RP/0/RSP0/CPU0:router(config-mon)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/11
RP/0/RSP0/CPU0:router(config-if)# ipv4 access-group span ingress
RP/0/RSP0/CPU0:router(config-if)# monitor-session ms1
RP/0/RSP0/CPU0:router(config-if-mon)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# ipv4 access-list span
RP/0/RSP0/CPU0:router(config-ipv4-acl)# 5 permit ipv4 any any dscp 5 capture
RP/0/RSP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 any any
RP/0/RSP0/CPU0:router(config-ipv4-acl)# commit
```

Layer 2 ACL-Based Traffic Mirroring: Example

The following example shows how to configure Layer 2 ACL-based traffic mirroring:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# monitor-session ms1
RP/0/RSP0/CPU0:router(config-mon)# destination interface gig0/2/0/15
```

```
RP/0/RSP0/CPU0:router(config-mon)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface gig0/2/0/11
RP/0/RSP0/CPU0:router(config-if)# l2transport
RP/0/RSP0/CPU0:router(config-if-l2)# exit
RP/0/RSP0/CPU0:router(config-if)# ethernet-services access-group acl_mirror ingress
RP/0/RSP0/CPU0:router(config-if)# acl
RP/0/RSP0/CPU0:router(config-if)# monitor-session ms1
RP/0/RSP0/CPU0:router(config-if-mon)# commit

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# ipv4 access-list acl_mirror
RP/0/RSP0/CPU0:router(config-ipv4-acl)# 5 permit ipv4 any any dscp 5 capture
RP/0/RSP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 any any
RP/0/RSP0/CPU0:router(config-ipv4-acl)# commit
```

Partial Packet Mirroring: Example

The following example shows how to configure mirroring of the first 100 bytes of the packet:

```
RP/0/RP0/CPU0:router(config)# interface gigabitethernet0/0/0/11
RP/0/RP0/CPU0:router(config-if-l2)# monitor-session mon1
RP/0/RSP0/CPU0:router(config-if-mon)# mirror first 100
```

Troubleshooting Traffic Mirroring

When you have issues with your traffic mirroring, begin your troubleshooting by checking the output of the **show monitor-session status** command. This command displays the recorded state of all sessions and source interfaces:

```
Monitor-session sess1
<Session status>
================================================================================
Source Interface      Dir   Status
--------------------- ----  ----------------------------------------------------
Gi0/0/0/0             Both  <Source interface status>
Gi0/0/0/2             Both  <Source interface status>
```

In the preceding example, the line marked as `<Session status>` can indicate one of the following configuration errors:

Session Status	Explanation
Session is not configured globally	The session does not exist in global configuration. Check **show run** command output to ensure that a session with the right name has been configured.
Destination interface <intf> is not configured	The interface that has been configured as the destination does not exist. For example, the destination interface may be configured to be a VLAN subinterface, but the VLAN subinterface may not have been created yet.

Session Status	Explanation
Destination interface <intf> (<down-state>)	The destination interface is not in Up state in the Interface Manager. You can verify the state using the **show interfaces** command. Check the configuration to see what might be keeping the interface from coming up (for example, a sub-interface needs to have an appropriate encapsulation configured).
Destination pseudowire is not configured	The L2VPN configuration that is to set up the pseudowire is missing. Configure the traffic mirroring session name as one segment of the xconnect p2p.
Destination pseudowire <name> (down)	The pseudowire is configured, but is down. Check the L2VPN configuration to identify why the pseudowire is not coming up.

The <Source interface status> can report the following messages:

Source Interface Status	Explanation
Operational	Everything appears to be working correctly in traffic mirroring PI. Please follow up with the platform teams in the first instance, if mirroring is not operating as expected.
Not operational (Session is not configured globally)	The session does not exist in global configuration. Check the **show run** command output to ensure that a session with the right name has been configured.
Not operational (destination interface not known)	The session exists, but it either does not have a destination interface specified, or the destination interface named for the session does not exist (for example, if the destination is a sub-interface that has not been created).
Not operational (source same as destination)	The session exists, but the destination and source are the same interface, so traffic mirroring does not work.
Not operational (destination not active)	The destination interface or pseudowire is not in the Up state. See the corresponding *Session status* error messages for suggested resolution.
Not operational (source state <down-state>)	The source interface is not in the Up state. You can verify the state using the **show interfaces** command. Check the configuration to see what might be keeping the interface from coming up (for example, a sub-interface needs to have an appropriate encapsulation configured).
Error: see detailed output for explanation	Traffic mirroring has encountered an error. Run the **show monitor-session status detail** command to display more information.

The **show monitor-session status detail** command displays full details of the configuration parameters, and of any errors encountered. For example:

```
RP/0/RSP0/CPU0:router#show monitor-session status detail

Monitor-session sess1
  Destination interface is not configured
  Source Interfaces
  -----------------
  GigabitEthernet0/0/0/0
    Direction: Both
    ACL match: Enabled
    Portion:   Full packet
    Status:    Not operational (destination interface not known).
  GigabitEthernet0/0/0/2
    Direction: Both
    ACL match: Disabled
    Portion:   First 100 bytes
    Status:    Error: 'Viking SPAN PD' detected the 'warning' condition 'PRM connection 
creation failure'.
```

This detailed output may give you a clear indication of what the problem is.

Here are additional trace and debug commands:

```
RP/0/RSP0/CPU0:router# show monitor-session platform trace ?

  all     Turn on all the trace
  errors  Display errors
  events  Display interesting events

RP/0/RSP0/CPU0:router# show monitor-session trace ?

  process  Filter debug by process

RP/0/RSP0/CPU0:router# debug monitor-session platform ?

  all     Turn on all the debugs
  errors  VKG SPAN EA errors
  event   VKG SPAN EA event
  info    VKG SPAN EA info

RP/0/RSP0/CPU0:router# debug monitor-session platform all

RP/0/RSP0/CPU0:router# debug monitor-session platform event

RP/0/RSP0/CPU0:router# debug monitor-session platform info

RP/0/RSP0/CPU0:router# show monitor-session status ?

  detail    Display detailed output
  errors    Display only attachments which have errors
  internal  Display internal monitor-session information
  |         Output Modifiers

RP/0/RSP0/CPU0:router# show monitor-session status

RP/0/RSP0/CPU0:router# show monitor-session status errors

RP/0/RSP0/CPU0:router# show monitor-session status internal
```

Where to Go Next

When you have configured an Ethernet interface, you can configure individual VLAN subinterfaces on that Ethernet interface.

For information about modifying Ethernet management interfaces for the shelf controller (SC), route processor (RP), and distributed RP, see the Advanced Configuration and Modification of the Management Ethernet Interface on the Cisco ASR 9000 Series Router module later in this document.

For information about IPv6 see the *Implementing Access Lists and Prefix Lists on Cisco IOS XR Software* module in the *Cisco IOS XR IP Addresses and Services Configuration Guide.*

Additional References

The following sections provide references related to implementing Gigabit and 10-Gigabit Ethernet interfaces.

Related Documents

Related Topic	Document Title
Ethernet L2VPN	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*
Cisco IOS XR master command reference	*Cisco ASR 9000 Series Aggregation Services Router Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*
Information about user groups and task IDs	*Cisco IOS XR Interface and Hardware Component Command Reference*

Standards

Standards	Title
IEEE 802.1ag ITU-T Y.1731	—

MIBs

MIBs	MIBs Link
IEEE CFM MIB	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/support

Configuring Virtual Loopback and Null Interfaces on the Cisco ASR 9000 Series Router

This module describes the configuration of loopback and null interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

Loopback and null interfaces are considered virtual interfaces.

A virtual interface represents a logical packet switching entity within the router. Virtual interfaces Interfaces have a global scope and do not have an associated location. Virtual interfaces have instead a globally unique numerical ID after their names. Examples are Loopback 0, Loopback 1Loopback1, and Loopback 99999. The ID is unique per virtual interface type to make the entire name string unique such that you can have both Loopback 0 and Null 0.

Loopback and null interfaces have their control plane presence on the active route switch processor (RSPRP). The configuration and control plane are mirrored onto the standby RSP RP and, in the event of a failoverswitchover, the virtual interfaces move to the ex-standby, which then becomes the newly active RSPRP.

Feature History for Configuring Loopback and Null Interfaces on Cisco IOS XR Software

Release	Modification
Release 3.7.2	This feature was introduced on the Cisco ASR 9000 Series Router.

Contents

Prerequisites for Configuring Virtual Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Configuring Virtual Interfaces

To configure virtual interfaces, you must understand the following concepts:

- Virtual Loopback Interface Overview, page 306
- Null Interface Overview, page 306
- Virtual Management Interface Overview, page 307
- Active and Standby RPs and Virtual Interface Configuration, page 307

Virtual Loopback Interface Overview

A virtual loopback interface is a virtual interface with a single endpoint that is always up. Any packet transmitted over a virtual loopback interface is immediately received by the selfsame interface. Loopback interfaces emulate a physical interface.

In Cisco IOS XR software, ,virtual loopback interfaces perform the following functions:

- Loopback loopback interfaces can act as a termination address for routing protocol sessions. This allows routing protocol sessions to stay up even if the outbound interface is down.
- You you can ping the loopback interface to verify that the router IP stack is working properly.

In applications where other routers or access servers attempt to reach a virtual loopback interface, you must configure a routing protocol to distribute the subnet assigned to the loopback address.

Packets routed to the loopback interface are rerouted back to the router or access server and processed locally. IP packets routed out the loopback interface but not destined to the loopback interface are dropped. Under these two conditions, the loopback interface can behave like a null interface.

Null Interface Overview

A null interface functions similarly to the null devices available on most operating systems. This interface is always up and can never forward or receive traffic; encapsulation always fails. The null interface provides an alternative method of filtering traffic. You can avoid the overhead involved with using access lists by directing undesired network traffic to the null interface.

The only interface configuration command that you can specify for the null interface is the **ipv4 unreachables** command. With the **ipv4 unreachables** command, if the software receives a nonbroadcast packet destined for itself that uses a protocol it does not recognize, it sends an Internet Control Message Protocol (ICMP) protocol unreachable message to the source. If the software receives a datagram that it cannot deliver to its ultimate destination because it knows of no route to the destination address, it replies to the originator of that datagram with an ICMP host unreachable message.

The Null 0 Null0 interface is created by default on the RSP RP during boot and cannot be removed. The **ipv4 unreachables** command can be configured for this interface, but most configuration is unnecessary because this interface just discards all the packets sent to it.

The Null 0 Null0 interface can be displayed with the **show interfaces null0** command.

Virtual Management Interface Overview

Configuring an IPv4 virtual address enables you to access the router from a single virtual address with a management network without prior knowledge of which RSP RP is active. An IPv4 virtual address persists across route switch processor (RSPRP) failover switchover situations. For this to happen, the virtual IPv4 address must share a common IPv4 subnet with a management Ethernet interface on both RPs.

On a Cisco ASR 9000 Series Router Cisco XR 12000 Series RouterCisco CRS-1 Router where each RSP RP has multiple management Ethernet interfaces, the virtual IPv4 address maps to the management Ethernet interface on the active RSP RP that shares the same IP subnet.

Active and Standby RPs and Virtual Interface Configuration

The standby RSP RP is available and in a state in which it can take over the work from the active RSP RP should that prove necessary. Conditions that necessitate the standby RSP RP to become the active RSP RP and assume the active RSPRP’s duties include:

- Failure detection by a watchdog
- Administrative command to take over
- Removal of the active RSP RP from the chassis

If a second RSP RP is not present in the chassis while the first is in operation, a second RSP RP may be inserted and automatically becomes the standby RSPRP. The standby RSP RP may also be removed from the chassis with no effect on the system other than loss of RSP RP redundancy.

After failoverswitchover, the virtual interfaces all are present on the standby (now active) RSPRP. Their state and configuration are unchanged and there has been no loss of forwarding (in the case of tunnels) over the interfaces during the failoverswitchover. The routers use nonstop forwarding (NSF) over bundles and tunnels through the failover switchover of the host RSPRP.

Note The user need not configure anything to guarantee that the standby interface configurations are maintained.

Note Protocol configuration such as tacacs source-interface, snmp-server trap-source, ntp source, logging source-interface do not use the virtual management IP address as their source by default. Use the **ipv4 virtual address use-as-src-addr** command to ensure that the protocol uses the virtual IPv4 address as its source address. Alternatively, you can also configure a loopback address with the designated or desired IPv4 address and set that as the source for protocols such as TACACS+ using the **tacacs source-interface** command.

How to Configure Virtual Interfaces

This section contains the following procedures:

- Configuring Virtual Loopback Interfaces, page 308 (Required)
- Configuring Null Interfaces, page 309 (Required)
- Configuring Virtual IPv4 IPV4 Interfaces, page 311 (Required)

Configuring Virtual Loopback Interfaces

This task explains how to configure a basic loopback interface.

Restrictions

The IP address of a loopback interface must be unique across all routers on the network. It must not be used by another interface on the router, and it must not be used by an interface on any other router on the network.

SUMMARY STEPS

1. **configure**
2. **interface loopback** *instance*
3. **interface loopback** *interface-path-id*
4. **ipv4 address** *ip-address*
5. **end**
   or
   commit
6. **show interfaces** *type instance*
7. **show interfaces** *type interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0RP00/CPU0:router# configure`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**`interface loopback`** *`instance`*
	`interface loopback` *`interface-path-id`*
	
	Example:
	`RP/0/RSP0RP00/CPU0:router#(config)# interface Loopback 3`
		Enters interface configuration mode and names the new loopback interface.

Step 3	**`ipv4 address`** *`ip-address`*
	
	Example:
	`RP/0/RSP0RP000/CPU0:router(config-if)# ipv4 address 172.18.189.38/32`
		Assigns an IP address and subnet mask to the virtual loopback interface using the **ipv4 address** configuration command.

Step 4	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0RP00/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0RP00/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show interfaces`** *`type instance`*
	`show interfaces` *`type interface-path-id`*
	
	Example:
	`RP/0/RSP0RP00/CPU0:router# show interfaces Loopback 3`
		(Optional) Displays the configuration of the loopback interface.

Configuring Null Interfaces

This task explains how to configure a basic null Null interface.

SUMMARY STEPS

1. **configure**
2. **interface null 0**

3. **end**
   or
   commit
4. **show interface null 0**
5. **show interfaces** *type interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0RP00/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface null 0`**
	Example:
	`RP/0/RSP0RP00/CPU0:router#(config)# interface null 0`
		Enters the null 0 null0 interface configuration mode.

Step 3	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0RP00/CPU0:router(config-null0)# end`
	or
	`RP/0/RSP0RP00/CPU0:router(config-null0)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 4	**`show interfaces null 0`**
	Example:
	`RP/0/RSP0RP00/CPU0:router# show interfaces null 0null0`
		Verifies the configuration of the null interface.

Configuring Virtual IPv4 IPV4 Interfaces

This task explains how to configure an IPv4 virtual interface.

SUMMARY STEPS

1. **configure**
2. **ipv4 address virtual address** *ipv4ip-address subnet mask*
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0RP00/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`ipv4 address virtual address`** *`ipv4-address subnet address/mask`*
	Example:
	`RP/0/RSP0RP00/CPU0:router(config)# ipv4 virtual address 10.3.32.154/8`
		Defines an IPv4 virtual address for the management Ethernet interface.

Step 3	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0RP00/CPU0:router(config-null0)# end`
	or
	`RP/0/RSP0RP00/CPU0:router(config-null0)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuration Examples for Virtual Interfaces

This section provides the following configuration examples:

- Configuring a Loopback Interface: Example, page 312
- Configuring a Null Interface: Example, page 313

Configuring a Loopback Interface: Example

The following example indicates how to configure a loopback interface:

```
RP/0/RSP0RP00/CPU0:router# configure
RP/0/RSP0RP00/CPU0:router(config)# interface Loopback 3
RP/0/RSP0RP00/CPU0:router(config-if)# ipv4 address 172.18.189.38/32
RP/0/RSP0RP00/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

```
RP/0/RSP0RP00/CPU0:router# show interfaces Loopback 3

Loopback3 is up, line protocol is up
  Hardware is Loopback interface(s)
  Internet address is 172.18.189.38/32
  MTU 1514 bytes, BW Unknown
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation Loopback,  loopback not set
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
```

Configuring a Null Interface: Example

The following example indicates how to configure a null interface:

```
RP/0/RSP0RP00/CPU0:router# configure
RP/0/RSP0RP00/CPU0:router(config)# interface Null 0
RP/0/RSP0RP00/CPU0:router(config-null0)# ipv4 unreachables
RP/0/RSP0RP00/CPU0:router(config-null0)# end
Uncommitted changes found, commit them? [yes]: yes
RP/0/RSP0RP00/CPU0:router# show interfaces Null 0

Null0 is up, line protocol is up
  Hardware is Null interface
  Internet address is Unknown
  MTU 1500 bytes, BW Unknown
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation Null,  loopback not set
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
```

Configuring a Virtual IPv4 Interface: Example

```
RP/0/RSP0RP00/CPU0:router# configure
RP/0/RSP0RP00/CPU0:router(config)# ipv4 virtual address 10.3.32.154/8
RP/0/RSP0RP00/CPU0:router(config-null0)# commit
```

Additional References

The following sections provide references related to loopback and null interface configuration.

Related Documents

Related Topic	Document Title
Cisco ASR 9000 Series Router master command reference	*Cisco ASR 9000 Series Router Master Commands List*
Cisco ASR 9000 Series Router interface configuration commands	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a Cisco ASR 9000 Series Router using the Cisco IOS XR Software.	*Cisco ASR 9000 Series Router Getting Started Guide*
Information about user groups and task IDs	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using the Cisco IOS XR Software.	*Cisco IOS XR Getting Started Guide*
Information about user groups and task IDs	*Cisco IOS XR Interface and Hardware Component Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router

This module describes the configuration of Channelized SONET/SDH on the Cisco ASR 9000 Series Aggregation Services Routers.

Feature History for Configuring Channelized SONET/SDH on Cisco IOS XR Software

Release	Modification
Release 3.9.0	Support for the following SPA was introduced on the Cisco ASR 9000 Series Router: • Cisco 2-Port Channelized OC-12/DS0 SPA
Release 4.0.0	Support for the following SPA was introduced on the Cisco ASR 9000 Series Router: • Cisco 1-Port Channelized OC-48/STM-16 SPA Support for SDH, E3, E1, and POS channelization was added for the Cisco 2-Port Channelized OC-12/DS0 and Cisco 1-Port Channelized OC-48/STM-16 SPAs.
Release 4.0.1	Support for the following SPA was introduced on the Cisco ASR 9000 Series Router: • Cisco 1-Port Channelized OC-3/STM-1 SPA

Contents

Prerequisites for Configuring Channelized SONET/SDH

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring Channelized SONET/SDH, be sure that the following tasks and conditions are met:

- You have at least one of the following SPAs installed in your chassis:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 2-Port Channelized OC-12c/DS0 SPA
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
- You should know how to apply and specify the SONET controller name and *interface-path-id* with the generalized notation *rack/slot/module/port*. The SONET controller name and *interface-path-id* are required with the **controller sonet** command.

Information About Configuring Channelized SONET/SDH

To configure Channelized SONET/SDH, you must understand the following concepts:

- Channelized SONET Overview, page 318
- Channelized SDH Overview, page 323
- Default Configuration Values for Channelized SONET/SDH, page 326
- How to Configure Channelized SONET/SDH, page 327

Channelized SONET Overview

Synchronous Optical Network (SONET) is an American National Standards Institute (ANSI) specification format used in transporting digital telecommunications services over optical fiber.

Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

Channelized SONET provides the ability to transport SONET frames across multiplexed T3/E3 and virtual tributary group (VTG) channels.

Channelized SONET is supported on the following SPAs:

- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 2-Port Channelized OC-12c/DS0 SPA

Channelized SDH is supported on the following SPAs:

- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 2-Port Channelized OC-12c/DS0 SPA

SONET uses Synchronous Transport Signal (STS) framing. An STS is the electrical equivalent to an optical carrier 1 (OC-1).

SDH uses Synchronous Transport Mode (STM) framing. An STM-1 is the electrical equivalent to 3 optical carrier 1s (OC-1s).

A channelized SONET interface is a composite of STS streams, which are maintained as independent frames with unique payload pointers. The frames are multiplexed before transmission.

When a line is channelized, it is logically divided into smaller bandwidth channels called *paths*. These paths carry the SONET payload. The sum of the bandwidth on all paths cannot exceed the line bandwidth.

When a line is not channelized, it is called *clear channel*, and the full bandwidth of the line is dedicated to a single channel that carries broadband services.

An STS stream can be channelized into the following types of channels:

- T3/E3
- VT1.5 mapped T1
- Packet over SONET/SDH (POS) (OC12 and OC48 only)

The T3/E3 channels can be channelized further into T1s, and the T1s can be channelized into time slots (DS0s), except on the 1-Port Channelized OC-48/STM-16 SPA, which does not support T1 or DS0s.

Channelizing a SONET line consists of two primary processes:

- Configuring the controller
- Configuring the interface into channelized paths

You configure the controller first by setting the mode of the STS path. The mode can be set to T3, or VT1.5-mapped T1, or POS, depending on your hardware support.

Note POS is supported only on the STS-3c and STS-12c paths on the Cisco 1-Port Channelized OC-12/DS0 SPA and on the STS-3c, STS-12c, and STS-48c paths on the Cisco 1-Port Channelized OC-48/STM-16 SPA.

When the mode is specified, the respective controller is created, and the remainder of the configuration is applied on that controller. For example, mode T3 creates a T3 controller. The T3 controller can then be configured to a serial channel, or it can be further channelized to carry T1s, and those T1s can be configured to serial interfaces.

Depending on the support for your installed SPA, each STS path can be independently configured into T3s, E3s, or VTGs, and so on.

Figure 22 shows an example of three STS paths for a SONET controller. However, the 2-Port Channelized OC-12/DS0 SPA supports up to 12 STS paths, and the 1-Port Channelized OC-48/STM-16 SPA supports up to 48 STS paths, but the 1-Port Channelized OC-48/STM-16 SPA does not support VTGs.

Figure 22 SONET Controller STS Paths

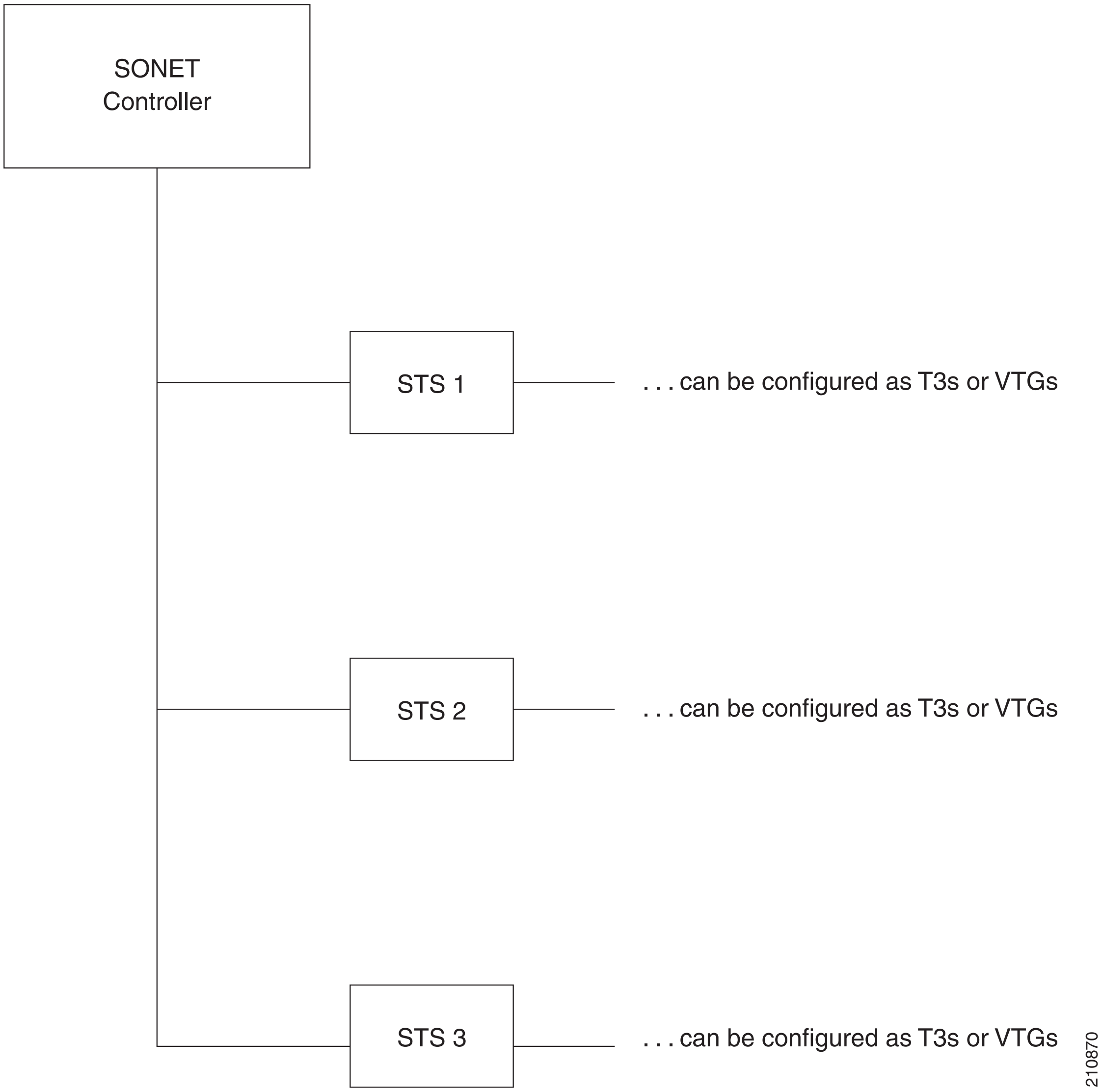

SONET
Controller
STS 1
. . . can be configured as T3s or VTGs
STS 2
. . . can be configured as T3s or VTGs
STS 3
. . . can be configured as T3s or VTGs
210870

Figure 23 shows an example of some SONET controller configuration combinations.

Note The 1-Port Channelized OC-48/STM-16 SPA on the Cisco ASR 9000 Series Router does not support VTGs.

Figure 23 SONET Controller Configuration Combinations

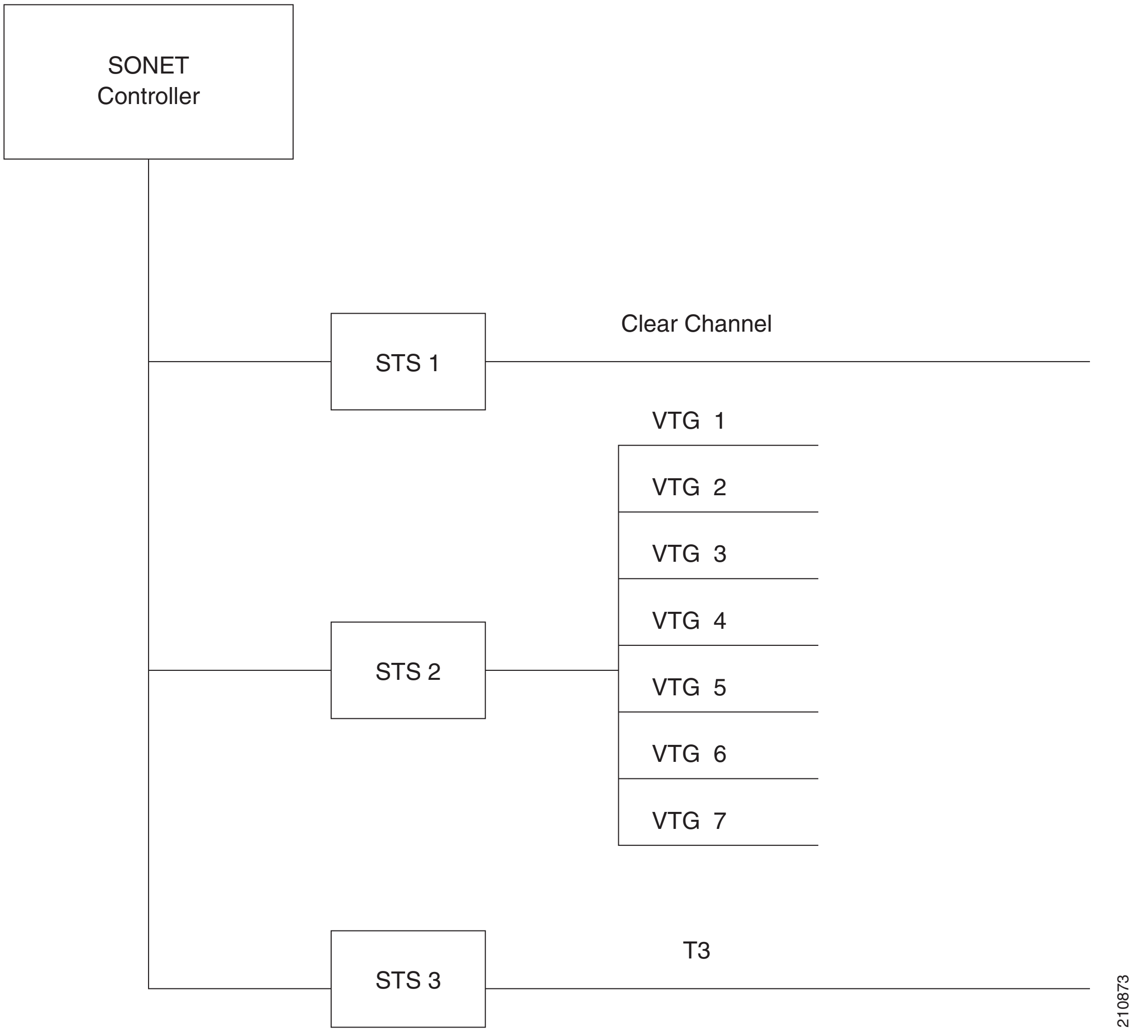


Figure 24 shows the T3 paths that can be configured.

Note Channelized T3 paths are only supported on the 1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA.

Figure 24 SONET T3 Channelized Paths

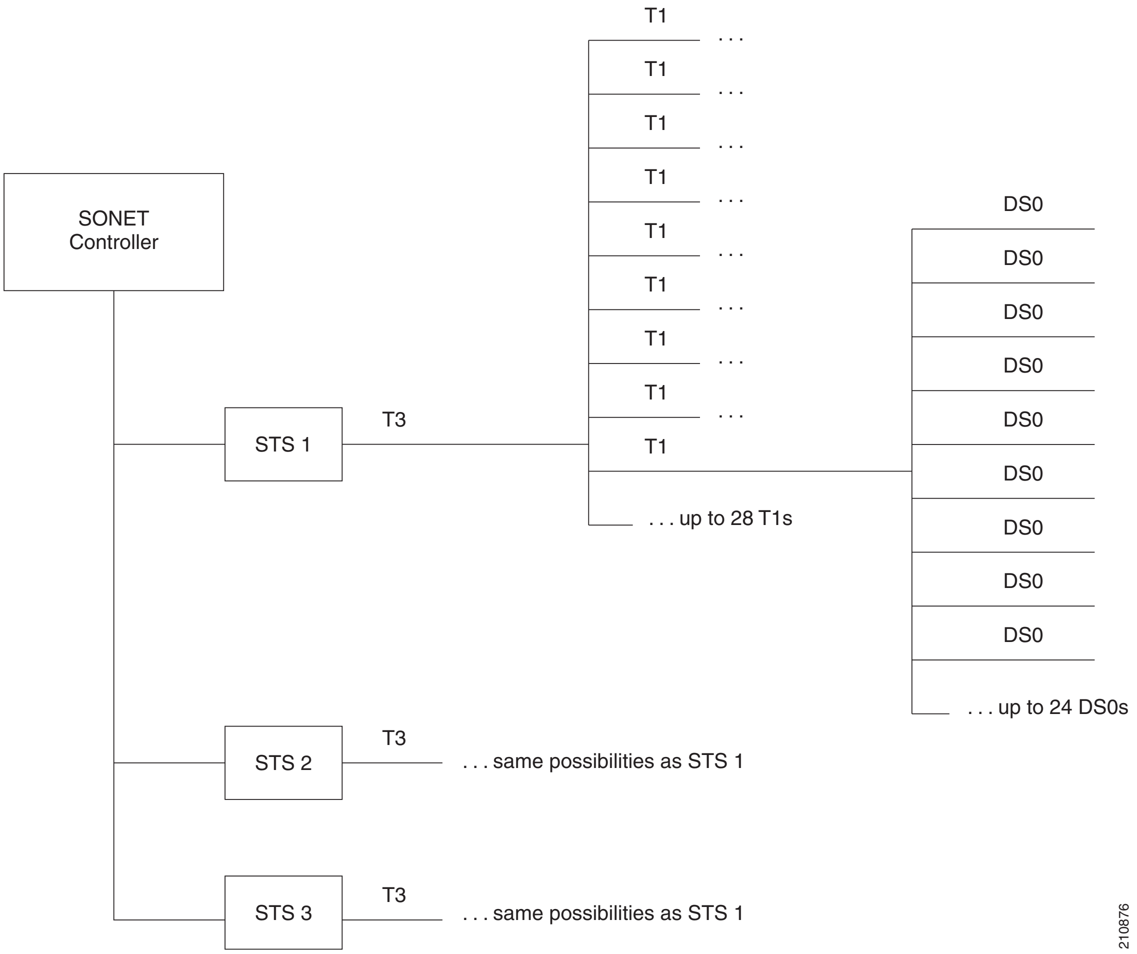
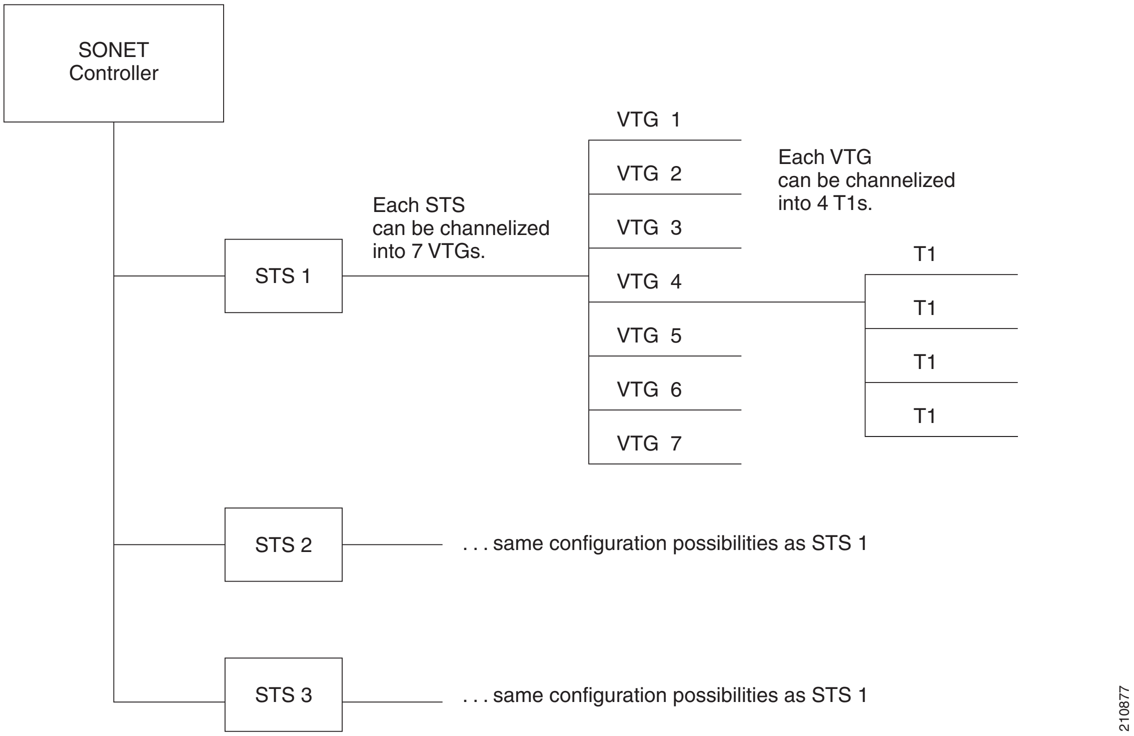


Figure 25 shows the VTG paths that can be configured.

Note VTG paths are only supported on the Cisco 1-Port Channelized OC-3/STM-1 SPA and Cisco 2-Port Channelized OC-12c/DS0 SPA on the Cisco ASR 9000 Series Router.

Figure 25 SONET VTG Channelized Paths


SONET Controller
STS 1
Each STS can be channelized into 7 VTGs.
VTG 1
VTG 2
VTG 3
VTG 4
VTG 5
VTG 6
VTG 7
Each VTG can be channelized into 4 T1s.
T1
T1
T1
T1
STS 2
. . . same configuration possibilities as STS 1
STS 3
. . . same configuration possibilities as STS 1
210877

Channelized SDH Overview

Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

Channelized SDH is supported on the following SPAs:

- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 2-Port Channelized OC-12/DS0 SPA

A Synchronous Transport Module (STM) signal is the Synchronous Digital Hierarchy (SDH) equivalent of the SONET STS, but the numbers are different for each bandwidth. In this guide, the STM term refers to both path widths and optical line rates. The paths within an STM signals are called administrative units (AUs).

A summary of the basic terminology differences between SONET and SDH is as follows:

- SONET STS is equivalent to SDH administrative unit (AU)
- SONET VT is equivalent to SDH tributary unit (TU)
- SDH basic building blocks are STM-1 (equivalent to STS-3) and STM-0 (equivalent to STS-1)

An administrative unit (AU) is the information structure that provides adaptation between the higher-order path layer and the multiplex section layer. It consists of an information payload (the higher-order virtual container) and an administrative unit pointer, which indicates the offset of the payload frame start relative to the multiplex section frame start.

An AU can be channelized into tributary units (TUs) and tributary unit groups (TUGs).

An administrative unit 4 (AU-4) consists of three STM-1s or an STM-3.

An administrative unit 3 (AU-3) consists of one STM-1.

An administrative unit group (AUG) consists of one or more administrative units occupying fixed, defined positions in an STM payload.

Table 2 SONET and SDH Terminology Equivalencies

SONET Term	SDH Term
SONET	SDH
STS-3c	AU-4
STS-1	AU-3
VT	TU
SPE	VC
Section	Regenerator Section
Line	Multiplex Section
Path	Path

On the Cisco ASR 9000 Series Router, the following levels of SDH channelization are supported:

- 1-Port Channelized OC-3/STM-1 SPA
  - AU4 to TUG-3 to TUG-2 to VC-12 to E1 to NxDS0
  - AU4 to TUG-3 to VC-3 to DS3 (Clear Channel)
  - AU4 to TUG-3 to VC-3 to E3 (Clear Channel)
  - AU3 to TUG-2 to VC-11 to DS1 to NxDS0
- 2-Port Channelized OC-12/DS0 SPA
  - AU-4-4c (VC-4-4c)
  - AU-4 (VC-4)
  - AU-4 to TUG-3 to VC-3 to DS3
  - AU-4 to TUG-3 to VC-3 to E3
  - AU-4 to TUG-3 to TUG-2 to VC-11 to T1 to NxDS0
  - AU-4 to TUG-3 to TUG-2 to VC-12 to E1to NxDS0
  - AU-3 to VC-3 to DS3
  - AU-3 to TUG-2 to VC-11 to T1 to NxDS0
  - AU-3 to TUG-2 to VC-12 to E1to NxDS0
  - AU-3 to VC-3 to E3
  - AU-3 to VC-3 to DS3 to T1 to NxDS0

  - AU-3 to VC-3 to DS3 to E1 to NxDS0
- 1-Port Channelized OC-48/STM-16 SPA
  - DS3
  - E3
  - AU-3 (VC-3)
  - AU-4 (VC-4)
  - AU-4-4c (VC-4-4c)
  - AU-4-16c (VC-4-16c)

Figure 26 shows an example of SDH AU-3 paths that can be configured on certain supported SPAs.

Note The 1-Port Channelized OC-48/STM-16 SPA does not support further channelization of AU-3 paths into T1s.

Figure 26 SDH AU3 Paths

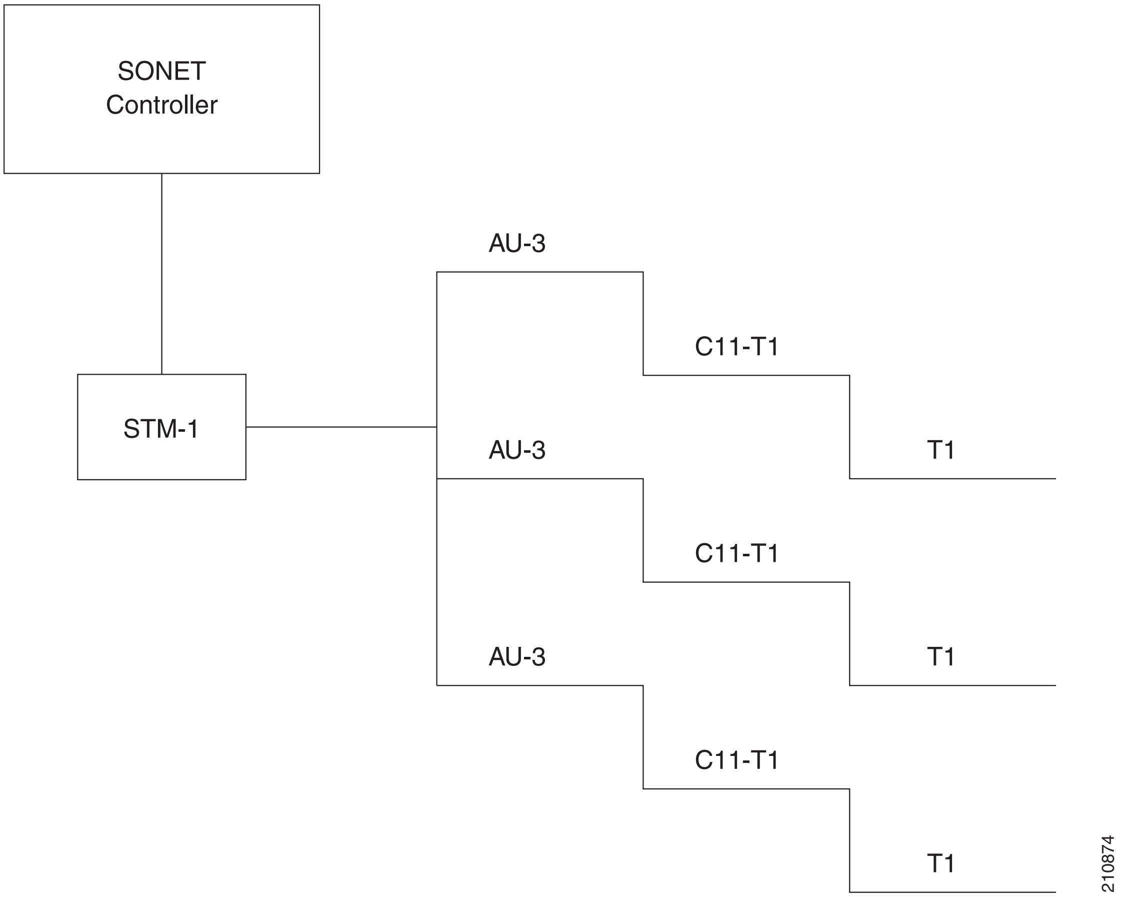
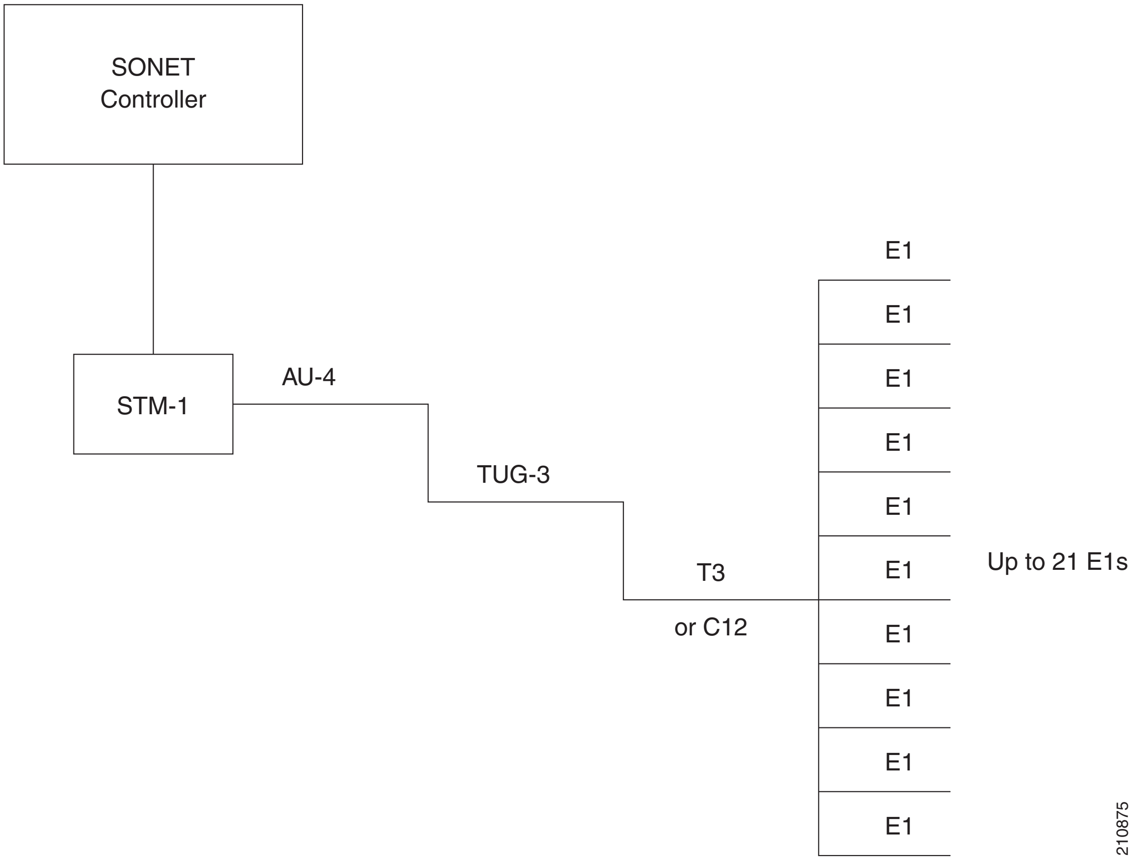


Figure 27 shows the SDH AU4 paths that can be configured on supported SPAs.

Note The 1-Port Channelized OC-48/STM-16 SPA only supports channelization to the T3 or E3 level. Further channelization of AU-4 paths is not supported.

Figure 27 SDH AU4 Paths



Default Configuration Values for Channelized SONET/SDH

Table 3 describes the default configuration parameters that are present on the Channelized SONET/SDH.

Table 3 SONET/SDH Controller Default Cit onfiguration Values

Parameter	Default Value	Configuration File Entry
Clock source	line	**clock source** {**internal** | **line**}
SONET framing	sonet	**framing** {**sdh** | **sonet**}

How to Configure Channelized SONET/SDH

This section contains the following procedures:

- Configuring SONET T3 and VT1.5-Mapped T1 Channels, page 327
- Configuring Packet over SONET Channels, page 332
- Configuring a Clear Channel SONET Controller for T3, page 335
- Configuring Channelized SONET APS, page 338
- Configuring SDH AU-3, page 341
- Configuring SDH AU-4, page 349

Configuring SONET T3 and VT1.5-Mapped T1 Channels

This task explains how to configure a SONET line into T3 and VT-mapped T1 Channels.

Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.
- STS paths can be channelized into T3s on the following SPAs:
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 2-Port Channelized OC-12/DS0 SPA
- STS paths can be channelized into VTG mapped T1s on the following SPA:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 2-Port Channelized OC-12/DS0 SPA
- T3 paths can be channelized into T1s or E1s on the following SPA:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 2-Port Channelized OC-12/DS0 SPA
- T1 paths can be channelized into NxDS0s on the Cisco 2-Port Channelized OC-12/DS0 SPA.

Restrictions

T1s and E1s are not supported on the Cisco 1-Port Channelized OC-48/STM-16 SPA.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sonet**

5. **sts** *number*
6. **mode** *mode*
7. **width** *number*
8. **root**
9. **controller** *controllerName instance*
10. **mode** *mode*
11. **root**
12. **controller t1** *interface-path-id*
13. **channel-group** *number*
14. **timeslots** *num1:num2:num3:num4*
    or
    timeslots *range1-range2*
15. **show configuration**
16. **root**
17. **interface serial** *interface-path-id*
18. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
19. **ipv4** *ip-address mask*
20. **no shutdown**
21. **end**
    or
    commit
22. **show**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/1/0`
		Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation.

	Command or Action	Purpose

Step 3	**clock source** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.
		• **line** is the default keyword.
		Note Internal clocking is required for SRP interfaces.

Step 4	**framing sonet** **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# framing sonet`	Configures the controller for SONET framing. SONET framing (**sonet**) is the default.
Step 5	**sts** *number* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# sts 1`	Configures the STS stream specified by *number.* The ranges are: • 1 to 48—1 Port Channelized OC-48/STM-16 SPA • 1 to 3—1-Port Channelized OC-3/STM-1 SPA • 1 to 12—2-Port Channelized OC-12/DS0 SPA
Step 6	**mode** *mode* **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# mode t3`	Sets the mode of interface at the STS level. The possible modes are: • t3—SONET path carrying T3 • vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1) (1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA only) • pos—Packet over SONET
Step 7	**width** *number* **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# width 3`	Configures the number of the STS streams that are concatenated. The possible values for *number* are: • 1—Indicating one STS stream • 3—Indicating three STS streams (STS-3c) • 12—Indicating concatenation of 12 STS streams (STS-12c) • 48—Indicating concatenation of 48 STS streams (STS-48c). This is the default on the 1-Port Channelized OC-48/STM-16 SPA. Widths 3, and 12, and 48 are configured on STS paths at natural boundaries, which coincide with the following path numbers: • 1, 4, 7, 10, and so on, for STS-3c • 1, 13, 25, and 37 for STS-12c • 1 for STS-48c
Step 8	**root** **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode.

	Command or Action	Purpose
Step 9	**controller** *controllerName instance* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t3 0/1/1/0/0`	Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/controllerName* notation. The controller names are: • t3—SONET path carrying T3 • vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1) (1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA only)
Step 10	**mode** *mode* **Example:** `RP/0/RSP0/CPU0:router(config-t3)# mode t1`	Sets the mode of interface at this level. The possible modes are: • t1—Channelized into 28 T1s (1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA only) • e1—Channelized into 21 E1s (1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA only) • serial—Clear channel carrying an HDLC-like payload
Step 11	**root** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 12	**controller t1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/1/1/0/0/0`	Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/T3Num/T1num* notation. (1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA only)
Step 13	**channel-group** *number* **Example:** `RP/0/RSP0/CPU0:router(config-t1)# channel-group 1`	Sets the channel group number to which time slots are assigned. The range is from 1 to 24.

Step 14	**timeslots** *num1:num2:num3:num4*
	or
	timeslots *range1-range2*
	Example:
	`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9`
	`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24`
		Specifies the time slots for the interface by number with the *num1:num2:num3:num4* notation, or by range with the *range1-range2* notation.

Step 15	**show configuration** **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# show configuration`	Displays the contents of uncommitted configuration.

	Command or Action	Purpose
Step 16	**root** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 17	**interface serial** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0`	Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

Step 18	**encapsulation** {**frame-relay** | **hdlc** | **ppp**}
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`
		Specifies the encapsulation type with the one of the following keywords:
		• **frame-relay**—Frame Relay network protocol
		• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol
		• **ppp**—Point-to-Point Protocol

Step 19	**ipv4** *ip-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255`	Assigns an IP address and subnet mask to the interface.
Step 20	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

	Command or Action	Purpose

Step 21	`end`
	or
	`commit`
	
	Example:
	`RP/0/0RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 22	**`show controllers sonet`** *`interface-path-id`*
	
	Example:
	`RP/0/RSP0/CPU0:router# show controllers sonet 0/1/1/0`
		Verifies the SONET controller configuration.

Configuring Packet over SONET Channels

This task explains how to configure Packet over SONET (POS) channels on SPAs supporting channelized SONET.

Prerequisites

You have one of the following SPAs installed:

- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 2-Port Channelized OC-12/DS0 SPA

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing** {**sdh** | **sonet**}
5. **sts** *number*
6. **width** *number*

7. **mode** *mode* **scramble**
8. **root**
9. **interface pos** *interface-path-id*
10. **encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]]
11. **pos crc** {**16** | **32**}
12. **mtu** *value*
13. **no shutdown**
14. **end**
or
commit
15. **show interfaces pos** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/1/0`
		Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation.

Step 3	**`clock source`** `{`**`internal`** `|` **`line`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.
		• **line** is the default keyword.
		Note Internal clocking is required for SRP interfaces.

Step 4	**`framing`** `{`**`sdh`** `|` **`sonet`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# framing sonet`
		(Optional) Configures the controller framing with either the **sdh** keyword for Synchronous Digital Hierarchy (SDH) framing or the **sonet** keyword for SONET framing.
		SONET framing (**sonet**) is the default.

Step 5	**`sts`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# sts 1`
		Configures the STS stream specified by *number*. The ranges are:
		• 1 to 12 on the 2-Port Channelized OC12c/DS0 SPA
		• 1 to 48 on the 1 Port Channelized OC48/DS3 SPA

	Command or Action	Purpose
Step 6	**width** *number* **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# width 3`	Configures the number of the STS streams that are concatenated. The possible values for *number* are: • 3—Indicating three STS streams (STS-3c) • 12—Indicating concatenation of 12 STS streams (STS-12c) • 48—Indicating concatenation of 48 STS streams (STS-48c) Widths 3, 12, and 48 are configured on STS paths at natural boundaries, which coincide with the following path numbers: • 1, 4, 7, 10, and so on, for STS-3c • 1, 13, 25, and 37 for STS-12c • 1 for STS-48c **Note** POS interfaces are not supported when width is 1.
Step 7	**mode** *mode* **scramble** **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# mode pos scramble`	Sets the mode of interface at the STS level. Set the mode to pos to create POS interface (OC12 and OC48 only).
Step 8	**root** **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode.
Step 9	**interface pos** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface POS 0/1/1/0`	Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 10	**encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]] **Example:** `RP/0/RSP0/CPU0:router(config-if)# encapsulation hdlc`	(Optional) Configures the interface encapsulation parameters and details such as HDLC or PPP. The default is HDLC.

Step 11	**pos crc** {**16** | **32**}
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# pos crc 32`
		(Optional) Configures the CRC value for the interface. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode.
		The default CRC is **32**.

Step 12	**mtu** *value* **Example:** `RP/0/RSP0/CPU0:router(config-if)# mtu 4474`	(Optional) Configures the POS MTU value. The range is 64–65535.

	Command or Action	Purpose
Step 13	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

Step 14	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 15	**show interfaces pos** *interface-path-id* **Example:** `RP/0/0/CPU0:router# show interfaces pos 0/1/1/0`	(Optional) Displays the interface configuration.

Configuring a Clear Channel SONET Controller for T3

This task explains how to configure a SONET line into a single T3 serial channel called *clear channel*. Clear channel is established by setting the T3 controller mode to serial.

Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}

4. **framing sonet**
5. **sts** *number*
6. **mode t3**
7. **root**
8. **controller t3** *interface-path-id*
9. **mode serial**
10. **root**
11. **interface serial** *interface-path-id*
12. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
13. **ipv4** *ip-address mask*
14. **no shutdown**
15. **end**
    or
    commit
16. **show controllers sonet** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/1/0`
		Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation.

Step 3	**`clock source`** `{`**`internal`** `|` **`line`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.
		• **line** is the default keyword.
		Note Internal clocking is required for SRP interfaces.

Step 4	**`framing sonet`**
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# framing sonet`
		Configures the controller for SONET framing. SONET framing (**sonet**) is the default.

	Command or Action	Purpose
Step 5	**sts** *number* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# sts 1`	Configures the STS stream specified by *number*. The ranges are: • 1 to 48—1-Port Channelized OC-48/DS3 SPA • 1 to 3—1-Port Channelized OC-3/STM-1 SPA • 1 to 12—2-Port Channelized OC-12/DS0 SPA
Step 6	**mode t3** **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# mode t3`	Sets the mode of the interface at the STS level for T3.
Step 7	**root** **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode.
Step 8	**controller t3** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t3 0/1/1/0/0`	Enters T3 controller configuration submode and specifies the T3 controller name and interface-path-id identifier with the *rack/slot/module/port/T3Num* notation.
Step 9	**mode serial** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# mode serial`	Sets the mode of the interface to serial to establish a clear channel.
Step 10	**root** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 11	**interface serial** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0`	Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

Step 12	**encapsulation** {**frame-relay** | **hdlc** | **ppp**}
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`
		Specifies the encapsulation type with the one of the following keywords:
		• **frame-relay**—Frame Relay network protocol
		• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol
		• **ppp**—Point-to-Point Protocol

Step 13	**ipv4** *ip-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255`	Assigns an IP address and subnet mask to the interface.

	Command or Action	Purpose
Step 14	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

Step 15	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 16	**show controllers sonet** *interface-path-id* **Example:** `RP/0//RSP0/CPU0:router# show controllers sonet 0/1/1/0`	Verifies the SONET controller configuration.

Configuring Channelized SONET APS

This task explains how to configure APS for channelized SONET lines.

Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.
- You should know how to configure the SONET APS as specified in the "Configuring SONET APS" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.

Restrictions

- SONET APS is not supported on the 1-Port Channelized OC-48/STM-16 SPA.
- The Cisco ASR 9000 Series Router supports multirouter APS only on the following SPAS:
  - 1-Port Channelized OC-3/STM-1 SPA
  - 2-Port Channelized OC-12c/DS0 SPA

SUMMARY STEPS

1. **aps group** *number*
2. **channel 0 local sonet** *interface*
   or
   channel 0 remote *ip-address*
3. **channel 1 local sonet** *interface*
   or
   channel 1 remote *ip-address*
4. **signalling** {**sonet** | **sdh**}
5. **end**
   or
   commit
6. **show aps**
7. **show aps group** [*number*]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`aps group`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# aps group 1`
		Adds an APS group with a specified number and enters APS group configuration mode.
		• Use the **aps group** command in global configuration mode.
		• To remove a group, use the **no** form of this command, as in: **no aps group** *number,* where the value range is from 1 to 255.
		Note To use the **aps group** command, you must be a member of a user group associated with the proper task IDs for APS commands.
		Note The **aps group** command is used even when a single protect group is configured.

Step 2	**`channel 0 local sonet`** *`interface`*
	`or`
	`channel 0 remote` *`ip-address`*
	Example:
	`RP/0/RSP0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1`
	`or`
	`RP/0/RSP0/CPU0:router(config-aps)# channel 0 remote 172.18.69.123`
		Creates a protect channel for the APS group, where **0** designates a protect channel.
		Note The protect channel must be assigned before the active channel can be assigned.
		Note To configure APS where both channels are on one router, use the **channel local** command for both the protect and active channels.
		To configure APS using two different routers where the active channel is on one router and the protect channel is on another router, use the **channel local** command for either the protect or the active channel, but use the **channel remote** command for the other channel.

Step 3	**`channel 1 local sonet`** *`interface`*
	`or`
	`channel 1 remote` *`ip-address`*
	Example:
	`RP/0/RSP0/CPU0:router(config-aps)# channel 1 local SONET 0/0/0/2`
	`or`
	`RP/0/0/CPU0:router(config-aps)# channel 1 remote 172.18.69.123`
		Creates an active channel for the APS group, where **1** designates an active channel.
		Note The active channel must be assigned after the protect channel is assigned.
		Note To configure APS where both channels are on one router, use the **channel local** command for both the protect and active channels.
		To configure APS using two different routers where the active channel is on one router and the protect channel is on another router, use the **channel local** command for either the protect or the active channel, but use the **channel remote** command for the other channel.

Step 4	**`signalling`** `{`**`sonet`** `|` **`sdh`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-aps)#` **`signalling sonet`**
		Configures the K1K2 overhead byte used for automatic protection switching (APS). The keyword options are:
		• **sonet**—Sets signaling to SONET.
		• **sdh**—Sets signaling to Synchronous Digital Hierarchy (SDH).

	Command or Action	Purpose

Step 5	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**`show aps`**
	Example:
	`RP/0/RSP0/CPU0:router# show aps`
		(Optional) Displays the operational status for all configured SONET APS groups.

Step 7	**`show aps group`** [*`number`*] **Example:** `RP/0/RSP0/CPU0:router# show aps group 3`	(Optional) Displays the operational status for configured SONET APS groups. **Note** The **show aps group** command is more useful than the **show aps** command when multiple groups are defined.

Configuring SDH AU-3

This section includes the following tasks:

- Configuring SDH AU-3 Mapped to C11-T1 or C12-E1, page 341
- Configuring SDH AU-3 Mapped to T3 or E3, page 345

Configuring SDH AU-3 Mapped to C11-T1 or C12-E1

This task explains how to configure SDH AU-3 with c11-t1 or c12-e1 mapping.

Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.

Restrictions

Channelized SDH AU-3 with c11-t1 or c12-e1 mapping is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 2-Port Channelized OC-12c/DS0 SPA

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sdh**
5. **au** *number*
6. **mode** *mode*
7. **root**
8. **controller t1** *interface-path-id*
9. **channel-group** *number*
10. **timeslots** *num1:num2:num3:num4*
    or
    timeslots *range1-range2*
11. **show configuration**
12. **root**
13. **interface serial** *interface-path-id*
14. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
15. **ipv4** *ip-address mask*
16. **no shutdown**
17. **end**
    or
    commit
18. **show controllers sonet** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/1/0`	Enters SONET controller configuration submode and specifies the SONET controller name and interface-path-id identifier with the *rack/slot/module/port* notation.

	Command or Action	Purpose

Step 3	**`clock source`** {**`internal`** | **`line`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.
		• **line** is the default keyword.
		Note Internal clocking is required for SRP interfaces.

Step 4	**`framing sdh`**
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# framing sdh`
		Configures the controller framing for Synchronous Digital Hierarchy (SDH) framing.
		SONET framing (**sonet**) is the default.

Step 5	**`au`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# au 1`
		Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is:
		• 1 to 3—1-Port Channelized OC-3/STM-1 SPA
		• 1 to 12—2-Port Channelized OC-12c/DS0 SPA
		Note The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4.

Step 6	**`mode`** *`mode`*
	Example:
	`RP/0/RSP0/CPU0:router(config-auPath)# mode c11-t1`
		Sets the mode of interface at the AU level. AU-3 paths can be mapped to c11-t1 or c12-e1 on supported SPAs.

Step 7	**`root`**
	Example:
	`RP/0/RSP0/CPU0:router(config-auPath)# root`
		Exits to global configuration mode.

Step 8	**`controller t1`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller T1 0/1/1/0/0/0/0`
		Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/t1Num* notation.

Step 9	**`channel-group`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# channel-group 0`
		Sets the channel-group number to which time slots are assigned. The range is from 1 to 28.

	Command or Action	Purpose

Step 10	**`timeslots`** *`num1:num2:num3:num4`*
	`or`
	`timeslots` *`range1-range2`*
	
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9`
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-12`
		Specifies time slots for the interface by number with the *num1:num2:num3:num4* notation, or by range with the *range1-range2* notation.

Step 11	**`show configuration`**
	
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# show configuration`
		Displays the contents of uncommitted configuration.

Step 12	**`root`**
	
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# root`
		Exits to global configuration mode.

Step 13	**`interface serial`** *`interface-path-id`*
	
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0`
		Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

Step 14	**`encapsulation`** `{`**`frame-relay`** `|` **`hdlc`** `|` **`ppp`**`}`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay`
		Specifies the encapsulation type with the one of the following keywords:
		• **frame-relay**—Frame Relay network protocol
		• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol
		• **ppp**—Point-to-Point Protocol

Step 15	**`ipv4`** *`ip-address mask`*
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255`
		Assigns an IP address and subnet mask to the interface.

Step 16	**`no shutdown`**
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration.
		Note Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

	Command or Action	Purpose

Step 17	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 18	**`show controllers sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show controllers sonet 0/1/1/0`
		Verifies the SONET controller configuration.

Configuring SDH AU-3 Mapped to T3 or E3

This task explains how to configure SDH AU-3 mapped to T3 or E3.

Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.

Restrictions

Channelized SDH AU-3 with T3 or E3 mapping is supported on the following SPAs:

- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 2-Port Channelized OC-12c/DS0 SPA

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}

4. **framing sdh**
5. **au** *number*
6. **mode t3**
   or
   mode e3
7. **root**
8. **controller** {**t3** | **e3**} *interface-path-id*
9. **mode serial**
10. **show configuration**
11. **root**
12. **interface serial** *interface-path-id*
13. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
14. **ipv4** *ip-address mask*
15. **no shutdown**
16. **end**
    or
    commit
17. **show controllers sonet** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/1/0`
		Enters SONET controller configuration submode and specifies the SONET controller name and interface-path-id identifier with the *rack/slot/module/port* notation.

Step 3	**`clock source`** `{`**`internal`** `|` **`line`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.
		• **line** is the default keyword.
		Note Internal clocking is required for SRP interfaces.

	Command or Action	Purpose
Step 4	**framing sdh** **Example:** RP/0/RSP0/CPU0:router(config-sonet)# framing sdh	Configures the controller framing for Synchronous Digital Hierarchy (SDH) framing. SONET framing (**sonet**) is the default.
Step 5	**au** *number* **Example:** RP/0/RSP0/CPU0:router(config-sonet)# au 1	Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is: • 1 to 48—1-Port Channelized OC-48/DS3 SPA • 1 to 3—1-Port Channelized OC-3/STM-1 SPA • 1 to 12—2-Port Channelized OC-12c/DS0 SPA **Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4.
Step 6	**mode t3** or **mode e3** **Example:** RP/0/RSP0/CPU0:router(config-auPath)# mode t3	Sets the mode of interface at the AU level to T3 or E3.
Step 7	**root** **Example:** RP/0/RSP0/CPU0:router(config-auPath)# root	Exits to global configuration mode.
Step 8	**controller** {**t3** | **e3**} *interface-path-id* **Example:** RP/0/RSP0/CPU0:router(config)# controller T3 0/1/1/0/0	Enters T3 or E3 controller configuration submode and specifies the T3 or E3 controller name and *interface-path-id* with the *rack/slot/module/port/auNum* notation.
Step 9	**mode serial** **Example:** RP/0/RSP0/CPU0:router(config-t3)# mode serial	Configures the mode of the port to be clear channel serial.
Step 10	**show configuration** **Example:** RP/0/RSP0/CPU0:router(config-t3)# show configuration	Displays the contents of uncommitted configuration.
Step 11	**root** **Example:** RP/0/RSP0/CPU0:router(config-t3)# root	Exits to global configuration mode.

	Command or Action	Purpose

Step 12	**`interface serial`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0`
		Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

Step 13	**`encapsulation frame-relay`** | **`hdlc`** | **`ppp`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay | hdlc | ppp`
		Specifies the encapsulation type with the one of the following keywords:
		• **frame-relay**—Frame Relay network protocol
		• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol
		• **ppp**—Point-to-Point Protocol

Step 14	**`ipv4`** *`ip-address mask`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255`
		Assigns an IP address and subnet mask to the interface.

Step 15	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration.
		Note Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

Step 16	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 17	**`show controllers sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show controllers sonet 0/1/1/0`
		Verifies the SONET controller configuration.

Configuring SDH AU-4

This task explains how to configure an SDH AU-4 stream into a TUG-3 channel mapped to E3s.

Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router* module.

Restrictions

- Channelized SDH is supported on the following SPAs:
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 2-Port Channelized OC-12/DS0 SPA
- In this release, AU-4 paths can only be channelized into TUG-3s.
- The 1-Port Channelized OC-48/STM-16 SPA does not support T1 or E1 channelization.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sdh**
5. **au** *number*
6. **mode tug3**
7. **width** *number*
8. **tug3** *number*
9. **mode** *mode*
10. **root**
11. **controller** *name interface-path-id*
12. **mode** *mode*
13. **root**
14. **controller** *name instance*
15. **channel-group** *number*
16. **timeslots** *num1:num2:num3:num4*
    or
    timeslots *range1-range2*
17. **show configuration**
18. **root**
19. **interface serial** *interface-path-id*

20. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
21. **ipv4** *ip-address mask*
22. **no shutdown**
23. **end**
or
commit
24. **show controllers sonet** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`controller sonet interface-path-id`
	Example:
	`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0`
		Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation.

Step 3	`clock source {internal | line}`
	Example:
	`RP/0/0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.
		• **line** is the default keyword.
		Note Internal clocking is required for SRP interfaces.

Step 4	`framing sdh`
	Example:
	`RP/0/0/CPU0:router(config-sonet)# framing sdh`
		Configures the controller for Synchronous Digital Hierarchy (SDH) framing.
		SONET framing (**sonet**) is the default.

Step 5	`au number`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# au 1`
		Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-4, the valid range is:
		• 1 to 16—1-Port Channelized OC-48/DS3 SPA
		• 1 to 3—1-Port Channelized OC-3/STM-1 SPA
		• 1 to 4—2-Port Channelized OC-12c/DS0 SPA
		Note The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4.

	Command or Action	Purpose

Step 6	**`mode tug3`**
	Example:
	`RP/0/0/CPU0:router(config-auPath)# mode tug3`
		Sets the mode of interface at the AU level. Currently only TUG3 is supported.

Step 7	**`width`** *`number`*
	Example:
	`RP/0/0/CPU0:router(config-auPath)# width 3`
		Configures the number of the AU streams.

Step 8	**`tug3`** *`number`*
	Example:
	`RP/0/0/CPU0:router(config-auPath)#tug3 1`
		Specifies the Tributary Unit Group (TUG) *number* and enters the config-tug3Path mode. The range is 1 to 3.

Step 9	**`mode`** *`mode`*
	Example:
	`RP/0/0/CPU0:router(config-tug3Path)# mode e3`
		Sets the mode of interface at the tug3 level. The modes are:
		• **c11**—TUG-3 path carrying TU-11
		• **c11-t1**—TUG-3 path carrying TU-11 to T1
		• **c12**—TUG-3 path carrying TU-12
		• **c12-e1**—TUG-3 path carrying TU-12 to E1
		• **e3**—TUG-3 path carrying E3
		• **t3**—TUG-3 path carrying T3
		Note The 1-Port Channelized OC-48/STM-16 SPA only supports the **e3** and **t3** options.

Step 10	**`root`**
	Example:
	`RP/0/0/CPU0:router(config-tug3Path)# root`
		Exits to global configuration mode.

Step 11	**`controller`** *`name instance`*
	Example:
	`RP/0/0/CPU0:router(config)# controller e3 0/1/1/0/0/0`
		Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/name/instance* notation. The controller names are:
		• **e3**—TUG3 path carrying E3
		• **t3**—TUG3 path carrying T3
		• **e1**—channelized E1 port
		Note In this step, you can create an E3 or T3 controller and add T1 channels under the T3 controller as shown in Step 14, or you can create a channelized E1 port at this point.
		Note E1 is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

	Command or Action	Purpose
Step 12	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-e3)#mode e1`	Sets the mode of interface. The modes are: • **e1**—Channelized into 21 E1s • **serial**—Clear Channel carrying HDLC-like payload • **t1**—Channelized into 28 T1s **Note** T1 and E1 are not supported on the 1-Port Channelized OC-48/STM-16 SPA.
Step 13	**root** **Example:** `RP/0/0/CPU0:router(config-e3)# root`	Exits to global configuration mode.
Step 14	**controller** *name instance* **Example:** `RP/0/0/CPU0:router(config)# controller E1 0/1/1/0/0/0/0/0`	Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/name/instance1/instance2* notation. The controller names are: • serial—Clear Channel carrying HDLC-like payload. • t1—Channelized into 24 T1s.
Step 15	**channel-group** *number* **Example:** `RP/0/0/CPU0:router(config-e1)# channel-group 0`	Sets the channel-group number to which time slots are assigned. • For t1, the range is from 1 to 24. • For e1, the range is from 1 to 32.

Step 16	**timeslots** *num1:num2:num3:num4*
	or
	timeslots *range1-range2*
	Example:
	`RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1:3:7:9`
	`RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1-12`
		Specifies time slots for the interface by number with the *num1:num2:num3:num4* notation, or by range with the *range1-range2* notation.

Step 17	**show configuration** **Example:** `RP/0/0/CPU0:router(config-e1-channel_group)# show configuration`	Displays the contents of uncommitted configuration.
Step 18	**root** **Example:** `RP/0/0/CPU0:router(config-e1-channel_group)# root`	Exits to global configuration mode.
Step 19	**interface serial** *interface-path-id* **Example:** `RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0`	Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

	Command or Action	Purpose

Step 20	**encapsulation** {**frame-relay** | **hdlc** | **ppp**}
	Example:
	`Router(config-if)# encapsulation frame-relay | hdlc | ppp`
		Specifies the encapsulation type with the one of the following keywords:
		• **frame-relay**—Frame Relay network protocol
		• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol
		• **ppp**—Point-to-Point Protocol

Step 21	**ipv4** *ip-address mask* **Example:** `Router(config-if)# ip address 10.10.10.10 255.255.255.255`	Assigns an IP address and subnet mask to the interface.
Step 22	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

Step 23	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-sonet)# end`
	or
	`RP/0/0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 24	**show controllers sonet** *interface-path-id* **Example:** `RP/0/0/CPU0:router# show controllers sonet 0/1/1/0`	Verifies the SONET controller configuration.

Configuration Examples for Channelized SONET

This section contains the following examples:

- Channelized SONET Examples, page 354
- Channelized SDH Examples, page 356

Channelized SONET Examples

- Channelized SONET T3 to T1 Configuration: Example, page 354
- Channelized Packet over SONET Configuration: Example, page 355
- SONET Clear Channel T3 Configuration: Example, page 355
- Channelized SONET APS Multirouter Configuration: Example, page 355

Channelized SONET T3 to T1 Configuration: Example

The following example shows SONET T3 to T1 configuration.

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode t3
  width 3
  root
 controller t3 0/1/1/0/0
  mode t1
  root
 controller t1 0/1/1/0/0/0
  framing esf
  channel-group 0
   timeslots 1:3:7:9
   show configuration
  root
 interface serial 0/1/1/0/0/0:0
 encapsulation hdlc
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

Channelized SONET in VT1.5 Mode and T1 Channelization to NxDS0

Note This example is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows how to configure SONET channelized to NxDS0s through SONET VT1.5 mode:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode vt15-t1
  root
controller t1 0/1/1/0/0/0
  channel-group 0 timeslots 1
  channel-group 1 timeslots 2-3
  commit
```

Channelized Packet over SONET Configuration: Example

The following example shows Channelized Packet over SONET configuration.

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode pos scramble
  width 3
  root
 interface POS 0/1/1/0
  encapsulation hdlc
  pos crc 32
  mtu 4474
  no shutdown
  commit
show interfaces pos 0/1/1/0
```

SONET Clear Channel T3 Configuration: Example

The following example shows SONET clear channel configuration for T3:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode t3
  root
 controller t3 0/1/1/0/0
  mode serial
  root
 interface serial 0/1/1/0/0/0:0
 encapsulation ppp
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

Channelized SONET APS Multirouter Configuration: Example

The following example shows SONET APS multirouter configuration.

```
 aps group 1
  channel 0 local SONET 0/0/0/1
  channel 1 remote 172.18.69.123
  signalling sonet
  commit
show aps
show aps group 3
```

Channelized SDH Examples

- Channelized SDH AU-3 Configuration: Examples, page 356
- Channelized SDH AU-4 Configuration: Examples, page 357

Channelized SDH AU-3 Configuration: Examples

This section includes the following configuration examples:

- Channelized SDH AU-3 to VC-3 and Clear Channel T3/E3: Examples, page 356
- Channelized SDH AU-3 to TUG-2, VC-11, T1 and NxDS0s: Example, page 356
- Channelized SDH AU-3 to TUG-2, VC-12, E1 and NxDS0s: Example, page 357

Channelized SDH AU-3 to VC-3 and Clear Channel T3/E3: Examples

The following example shows how to configure SDH AU-3 to VC-3 and clear channel T3:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  width 1
  mode t3
  root
 controller t3 0/1/1/0/1
  mode serial
  commit
```

The following example shows how to configure SDH AU-3 to VC-3 and clear channel E3:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  width 1
  mode e3
  root
 controller e3 0/1/1/0/1
  mode serial
  commit
```

Channelized SDH AU-3 to TUG-2, VC-11, T1 and NxDS0s: Example

Note This example is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows how to configure SDH AU-3 to TUG-2, VC-11 and channelized T1 to NxDS0s:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  mode c11-t1
  width 1
  root
 controller T1 0/1/1/0/0/1/1
  channel-group 0
   timeslots 1-12
   show configuration
  root
 interface serial 0/1/1/0/1/1:0
 encapsulation ppp
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

Channelized SDH AU-3 to TUG-2, VC-12, E1 and NxDS0s: Example

Note This example is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows how to configure SDH AU-3 to TUG-2, VC-12 and channelized E1 to NxDS0s:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  mode c12-e1
  width 1
  root
 controller e1 0/1/1/0/0/1/1
  channel-group 0
   timeslots 1-12
   show configuration
  root
 interface serial 0/1/1/0/1/1:0
 encapsulation ppp
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

Channelized SDH AU-4 Configuration: Examples

This section includes the following configuration examples:

- Channelized SDH AU-4 to TUG-3 and Clear Channel T3/E3: Examples, page 357
- Channelized SDH AU-4 to TUG-3, TUG-2, and T1/E1 and NxDS0: Examples, page 358

Channelized SDH AU-4 to TUG-3 and Clear Channel T3/E3: Examples

The following exampe shows SDH AU-4 channelization to TUG-3 and clear channel T3:

```
configure
 controller sonet 0/4/0/0
  framing sdh
  au 1
  width 3
  mode tug3
  tug3 1
   mode t3
   root
 controller t3 0/4/0/0/1/1
  mode serial
  commit
```

The following exampe shows SDH AU-4 channelization to TUG-3 and clear channel E3:

```
configure
 controller sonet 0/4/0/0
  framing sdh
  au 1
  width 3
  mode tug3
  tug3 1
   mode e3
   root
 controller e3 0/4/0/0/1/1
  mode serial
  commit
```

Channelized SDH AU-4 to TUG-3, TUG-2, and T1/E1 and NxDS0: Examples

Note Channelization to T1/E1 and NxDS0s is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows SDH AU-4 configuration with unframed E1 controllers and serial interfaces:

```
configure
 controller sonet 0/1/2/0
  framing sdh
  au 1
  width 3
  mode tug3
  tug3 1
   mode c12-e1
!
  tug3 2
   mode c12-e1
!
  tug3 3
   mode c12-e1
!
controller E1 0/1/2/0/1/1/1/1
framing unframed
!
controller E1 0/1/2/0/1/1/1/2
framing unframed
!
controller E1 0/1/2/0/1/1/1/3
framing unframed
!
interface Serial0/1/2/0/1/1/1/1:0
encapsulation ppp
multilink
  group 1
!
interface Serial0/1/2/0/1/1/1/2:0
encapsulation ppp
```

```
multilink
  group 1
!
!
interface Serial0/1/2/0/1/1/1/3:0
encapsulation ppp
multilink
  group 1
!
```

The following example shows SDH AU-4 configuration with E1 controller channel groups and serial interfaces:

```
configure
 controller SONET0/3/2/0
   framing sdh
   au 1
   width 3
   mode tug3
   tug3 1
   mode c12-e1
!
   tug3 2
   mode c12-e1
!
   tug3 3
   mode c12-e1
!
controller E1 0/3/2/0/1/1/1/1
 framing crc4
 channel-group 0
  timeslots 1-4
!
controller E1 0/3/2/0/1/1/3/1
 framing crc4
 channel-group 0
  timeslots 1-31
!
controller E1 0/3/2/0/1/1/1/2
 framing crc4
 channel-group 0
  timeslots 1-31
!
controller E1 0/3/2/0/1/2/7/3
 framing crc4
 channel-group 0
  timeslots 1-5
!
 channel-group 1
  timeslots 6-31
!
interface Serial0/3/2/0/1/1/1/1:0
 encapsulation frame-relay IETF
 frame-relay lmi-type ansi
 frame-relay intf-type dce
!
interface Serial0/3/2/0/1/1/1/1:0.1 point-to-point
 ipv4 address 192.168.200.2 255.255.255.252
 ipv4 verify unicast source reachable-via rx
 pvc 100
  encap ietf
!
interface Serial0/3/2/0/1/1/3/1:0
 encapsulation ppp
 multilink
```

```
  group 1
!
interface Serial0/3/2/0/1/1/1/2:0
 encapsulation ppp
 multilink
  group 1
```

Additional References

The following sections provide references related to channelized SONET configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using the Cisco IOS XR software	*Cisco IOS XR Getting Started Guide*
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
• CISCO-SONET-MIB • ENTITY-MIB • SONET-MIB (RFC 3592) The following additional MIBs are supported on the Cisco 1-Port Channelized OC-3/STM-1 SPA and Cisco 2-Port Channelized OC-12c/DS0 SPA on the Cisco ASR 9000 Series Router: • CISCO-IF-EXTENSION-MIB • DS1-MIB • DS3-MIB • IF-MIB	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Circuit Emulation over Packet on the Cisco ASR 9000 Series Router

This module describes the configuration of Circuit Emulation over Packet (CEoP) shared port adapters (SPAs) on the Cisco ASR 9000 Series Aggregation Services Routers.

Feature History for Configuring CEoP on Cisco ASR 9000 Series Router

Release	Modification
Release 4.2.0	• Support for Circuit Emulation Service over Packet Switched Network was added in the following SPA: – Cisco 1-port Channelized OC3/STM-1 SPA (SPA-1CHOC3-CE-ATM)

Contents

Prerequisites for Configuration

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring the Circuit Emulation over Packet (CEoP) service on your router, ensure that these conditions are met:

- You must have this SPAinstalled in your chassis:
  - Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and ATM SPA
- You should know how to apply and specify the SONET controller name and *interface-path-id* with the generalized notation *rack/slot/module/port*. The SONET controller name and *interface-path-id* are required with the **controller sonet** command.

Overview of Circuit Emulation over Packet Service

Circuit Emulation over Packet (CEoP) is a way to carry TDM circuits over packet switched network. Circuit Emulation over Packet is the imitation of a physical connection. The goal of CEoP is to replace leased lines and legacy TDM networks. This feature allows network administrators to use their existing IP/MPLS network to provide leased-line emulation services or to carry data streams or protocols that do not meet the format requirements of other multiservice platform interfaces. CEoP puts TDM bits into the packets, encapsulates them into appropriate header and then sends through PSN. The receiver side of CEoP restores the TDM bit stream from packets.

CEoP SPAs are half-height (HH) Shared Port Adapters (SPA) and the CEoP SPA family consists of 24xT1/E1/J1, 2xT3/E3, and 1xOC3/STM1 unstructured and structured (NxDS0) quarter rate, half height SPAs. The CEoP SPAs provide bit-transparent data transport that is completely protocol independent.

CEoP has two major modes:

- Unstructured mode is called SAToP (Structure Agnostic TDM over Packet) — SAToP does not look what is inside the incoming data and considers it as a pure bit stream.
- Structured mode is named CESoPSN (Circuit Emulation Service over Packet Switched Network) — CESoPSN is aware of the structure of the incoming TDM bit stream at DS0 level.

CESoPSN and SAToP can use MPLS, UDP/IP, and L2TPv3 for the underlying transport mechanism.

Note The Cisco IOS XR Release 4.2.x supports only MPLS transport mechanism.

These SPAs are the first Cisco router interfaces designed to meet the emerging standards for Circuit Emulation Services over Packet Switched Network (CESoPSN) and Structure-Agnostic Transport over Packet (SAToP) transport.

The Cisco IOS XR Release 4.2.x supports CEM functionality only on this SPA:

1-port Channelized OC3 SPA [SPA-1xOC3-CE-ATM]

In SAToP mode, these SPAs do not assume that data has any predefined format or structure. They simply regard the data as an arbitrary bit stream. All data bits are simply transported to a defined destination encapsulated in IP/MPLS packets. In CESoPSN mode the carrier has defined format. The SPAs support a full range of E1 and T1 framing. CESoPSN applications can save utilized bandwidth by selecting only valid timeslots for transmission. Some primary applications include:

- Transporting 2G and 3G network traffic over packet networks, for mobile operators. Mobile service providers are implementing high-speed data networks with HSDPA to support new revenue-generating services. The SPA is uniquely positioned for multigenerational migration of mobile networks (2G and 3G), simultaneously carrying TDM and ATM traffic over IP/MPLS networks. This technology provides a mechanism to enable IP/MPLS to the cell site, which can eventually be in place to transport the mobile traffic over IP from end to end.

- T3/E3 circuit emulation for leased-line replacement.
- T1/E1 circuit emulation for leased-line replacement.
- PBX to PBX connectivity over PSN.
- High density SS7 backhaul over IP/MPLS.
- Inter-MSC connectivity.
- Preencrypted data for government, defense, or other high-security applications.
- Proprietary synchronous or asynchronous data protocols used in transportation, utilities, and other industries.
- Leased-line emulation service offerings in metropolitan (metro) Ethernet or WAN service provider environments.

For more information on Circuit Emulation service concepts, configuration, and example, see the *Implementing Point to Point Layer 2 Services* module in the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*.

Information About Configuring CEoP Channelized SONET/SDH

To configure the Circuit Emulation over Packet Channelized SONET/SDH, you must understand the following concepts:

- Channelized SONET and SDH Overview, page 365
- Default Configuration Values for Channelized SONET/SDH, page 369

Channelized SONET and SDH Overview

Synchronous Optical Network (SONET) is an American National Standards Institute (ANSI) specification format used in transporting digital telecommunications services over optical fiber.

Channelized SONET provides the ability to transport SONET frames across multiplexed T3/E3 and virtual tributary group (VTG) channels.

SONET uses Synchronous Transport Signal (STS) framing. An STS is the electrical equivalent to an optical carrier 1 (OC-1).

A channelized SONET interface is a composite of STS streams, which are maintained as independent frames with unique payload pointers. The frames are multiplexed before transmission.

When a line is channelized, it is logically divided into smaller bandwidth channels called *paths*. These paths carry the SONET payload. The sum of the bandwidth on all paths cannot exceed the line bandwidth.

When a line is not channelized, it is called *clear channel*, and the full bandwidth of the line is dedicated to a single channel that carries broadband services.

The T3/E3 channels can be channelized into T1s, and the T1s can be channelized further into DS0 time slots.

Channelizing a SONET line consists of two primary processes:

- Configuring the controller
- Configuring the interface into channelized paths

You configure the controller first by setting the mode of the STS path.

When the mode is specified, the respective controller is created, and the remainder of the configuration is applied on that controller. For example, mode T3 creates a T3 controller. The T3 controller can then be configured to a serial channel, or it can be further channelized to carry T1s, and those T1s can be configured to serial interfaces.

Depending on the support for your installed SPA, each STS path can be independently configured into T3s, E3s, or VTGs, and so on.

The following level of SONET channelization modes are supported in CEoP SPA:

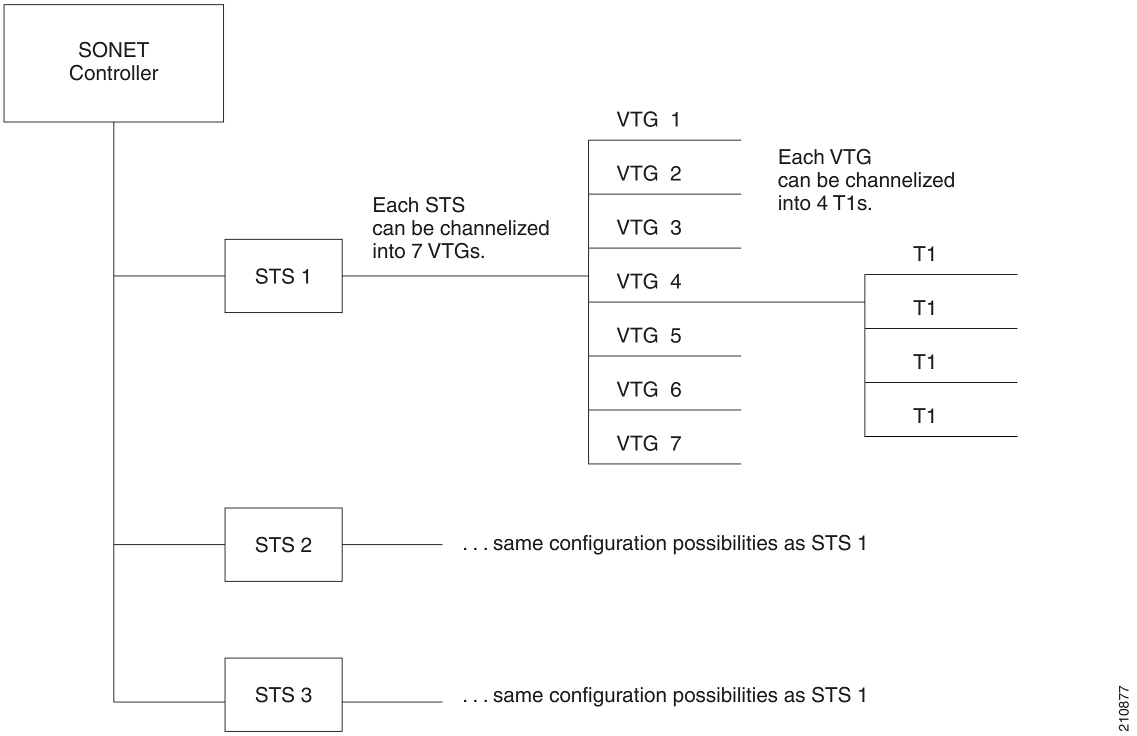
OC3->STS-1->VTG-> VT1.5 -> Unframed T1

OC3->STS-1->VTG-> VT1.5 -> T1 -> DS0

Figure 28 shows the VTG paths that can be configured.

Note Only VTG paths are supported on the Cisco 1-Port Channelized OC-3/STM-1 SPA on the Cisco ASR 9000 Series Router.

Figure 28 SONET VTG Channelized Paths



Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

SDH uses Synchronous Transport Mode (STM) framing. An STM-1 is the electrical equivalent to 3 optical carrier 1s (OC-1s). A Synchronous Transport Module (STM) signal is the Synchronous Digital Hierarchy (SDH) equivalent of the SONET STS, but the numbers are different for each bandwidth. In this guide, the STM term refers to both path widths and optical line rates. The paths within an STM signals are called administrative units (AUs).

A summary of the basic terminology differences between SONET and SDH is as follows:

- SONET STS is equivalent to SDH administrative unit (AU)
- SONET VT is equivalent to SDH tributary unit (TU)
- SDH basic building blocks are STM-1 (equivalent to STS-3) and STM-0 (equivalent to STS-1)

An administrative unit (AU) is the information structure that provides adaptation between the higher-order path layer and the multiplex section layer. It consists of an information payload (the higher-order virtual container) and an administrative unit pointer, which indicates the offset of the payload frame start relative to the multiplex section frame start.

An AU can be channelized into tributary units (TUs) and tributary unit groups (TUGs).

An administrative unit 3 (AU-3) consists of one STM-1.

An administrative unit group (AUG) consists of one or more administrative units occupying fixed, defined positions in an STM payload.

The Table 4 shows the commonly used notations and terms in SONET standards and their SDH equivalents.

Table 4 SONET and SDH Terminology Equivalencies

SONET Term	SDH Term
SONET	SDH
STS-3c	AU-4
STS-1	AU-3
VT	TU
SPE	VC
Section	Regenerator Section
Line	Multiplex Section
Path	Path

The following levels of SDH channelization are supported on the CEoP SPA in Cisco IOS XR Release 4.2.0:

- For E1 :
  - STM1-> AU-4 -> TUG-3 -> TUG-2 ->VC12-> Unframed E1
  - STM1-> AU-4 -> TUG-3 -> TUG-2 ->VC12-> E1 -> DS0
- For T1 :
  - STM1-> AU-3-> TUG-2 -> VC11->Unframed T1
  - STM1-> AU-3-> TUG-2 -> VC11->T1 -> DS0

Figure 29 shows an example of SDH AU-3 paths that can be configured on the CEoP SPA.

Figure 29 SDH AU3 Paths

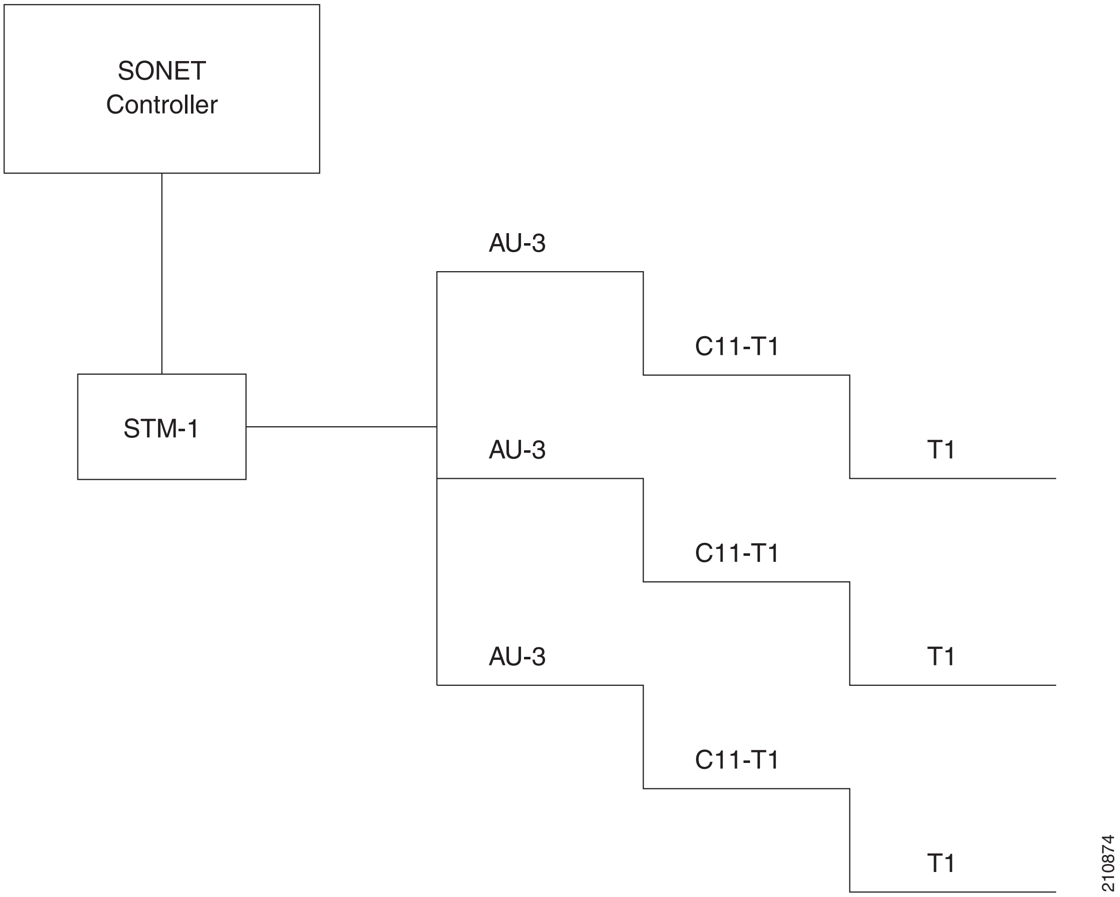
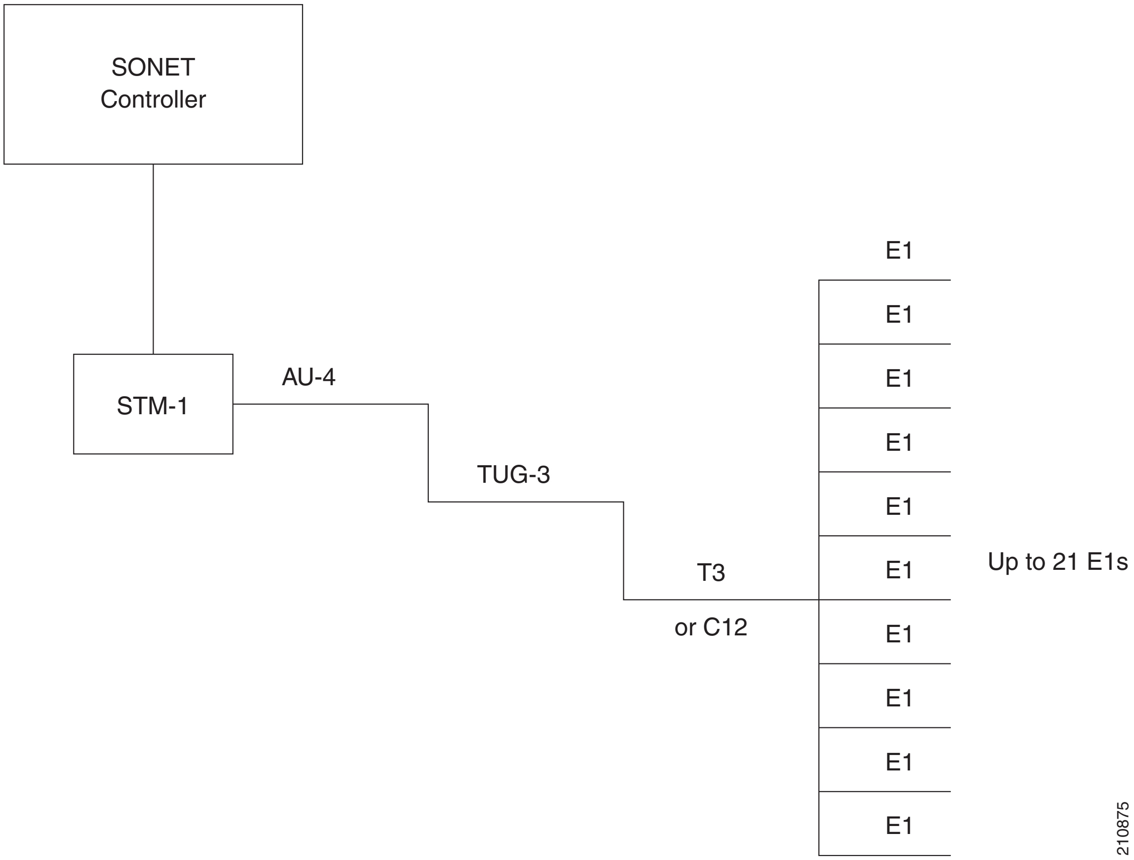


Figure 30 shows the SDH AU4 paths that can be configured on the CEoP SPA.

Figure 30 SDH AU4 Paths



Default Configuration Values for Channelized SONET/SDH

Table 5 describes the default configuration parameters that are present on the Channelized SONET/SDH.

Table 5 SONET/SDH Controller Default Cit onfiguration Values

Parameter	Default Value	Configuration File Entry
Clock source	line	**clock source** {**internal** | **line**}
SONET framing	sonet	**hw-module sub-slot** ***node-id*** **cardtype** {**sonet** | **sdh**}

Clock Distribution

Clocking distribution in the CEoP SPA can be done in these ways:

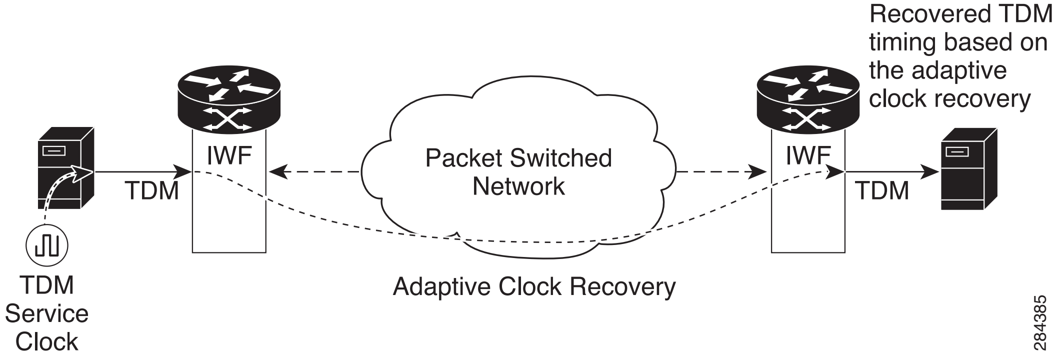
- Synchronous Clocking — With synchronous clocking, TDM lines on source and destination are synchronized to the same clock delivered by some means of physical clock distribution (SONET/SDH, BITS, GPS, and so on). The clock to the particular TDM line can be delivered from
  - Line: the transmit clock is from the receiver of the same physical line
  - Internal: the transmit clock is taken from line card and can be derived either from an internal free running oscillator or from another physical line
  - Recovered: In-band pseudowire-based activeclock recovery on a CEM interface which is used to drive the transmit clock.

  The number of recovered clocks that can be configured for CEoP SPA are:
  - Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA : 10 clocks per SPA in the T1/E1 mode.
- Adaptive Clocking — Adaptive clocking is used when the routers do not have a common clock source. See Figure 31. The clock is derived based on packet arrival rates. Two major types of adaptive clock recovery algorithms are:
  - Based on dejitter buffer fill level
  - Based on packet arrival rate

The clock quality depends on packet size, has less tolerance to packet loss/corruption and introduces unnecessary delay in order to have sufficient number of packets in the buffer for clock recovery. The dejitter buffer size determines the ability of the emulated circuit to tolerate network jitter. The dejitter buffer in CEoP software is configurable up to a maximum of 500 milliseconds.

Note The CEoP SPA hardware supports only the packet arrival rate algorithm.

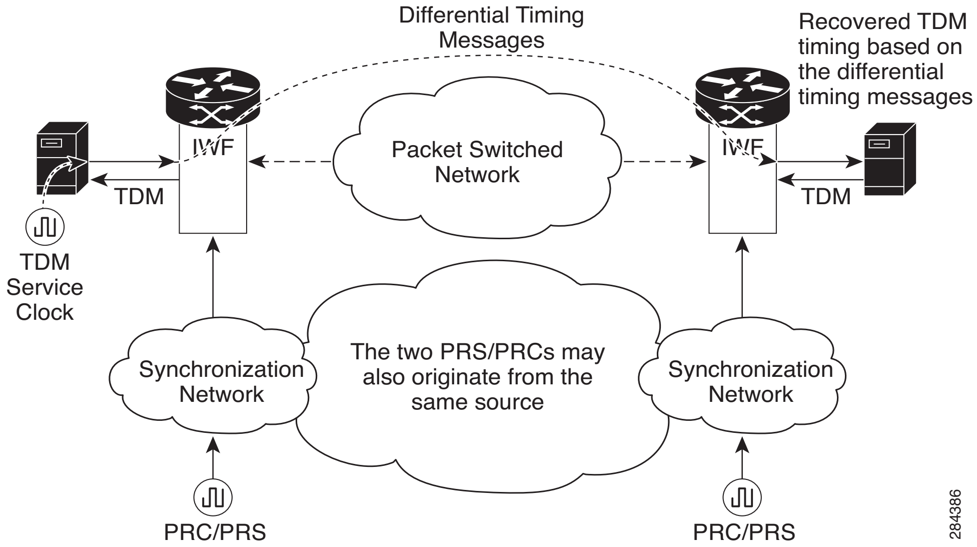
Figure 31 Adaptive Clock Recovery



- Differential clocking — Differential clocking is used when the cell site and aggregation routers have a common clock source but TDM lines are clocked by a different source. The TDM clocks are derived from differential information in the RTP header of the packet with respect to the common clock. Differential clock recovery is based on time stamps received in RTP header. On the master side, the difference of TDM clock and network clock is recorded into RTP header. On the slave side, these timestamps are read from RTP header, the clock recovery is done and this clock is used for synchronization. See Figure 32.

Note The Cisco 1-port Channelized OC3/STM-1 CEoP SPA hardware can recover only a maximum of ten unique clocks in as many CEM interfaces. The CEM interfaces where clock recovery is configured must be on unique T1s.

Figure 32 Differential Clock Recovery


Differential Timing Messages
Recovered TDM timing based on the differential timing messages
IWF
Packet Switched Network
IWF
TDM
TDM
TDM Service Clock
Synchronization Network
The two PRS/PRCs may also originate from the same source
Synchronization Network
PRC/PRS
PRC/PRS
284386

For information on CEM configuration and commands, see *Implementing Point to Point Layer 2 Services module* in the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* and *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*.

For a sample CEM interface configuration, refer Circuit Emulation Interface Configuration: Examples, page 392.

How to implement CEM

This section contains the following procedures:

- Configuring SONET VT1.5-Mapped T1 Channels and Creating CEM Interface, page 372
- Configuring SDH AU-3 Mapped to C11-T1 or C12-E1, page 375
- Configuring CEM Interface, page 381
- Configuring Clocking, page 389

Configuring SONET VT1.5-Mapped T1 Channels and Creating CEM Interface

In the case of Cisco 1-port Channelized OC3/STM-1 CEoP SPA, the STS stream can be channelized into the VT1.5 mapped T1 channel.

This task explains how to configure a SONET line into VT-mapped T1 Channels.

Prerequisites

None.

Restrictions

Channelized SONET STS stream with VT1.5-T1 mapping is supported on the following SPA:

- Cisco 1-Port Channelized OC-3/STM-1 SPA

SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*
3. **commit**
4. **controller sonet** *interface-path-id*
5. **sts** *number*
6. **mode** *mode*
7. **root**
8. **controller t1** *interface-path-id*
9. **cem-group unframed**
10. **controller t1** *interface-path-id*
11. **cem-group framed** *group-number* **timeslots** *range1-range2*
12. **no shutdown**
13. **end**
    or
    commit
14. **show runn interface cem** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**hw-module subslot** *node-id* **cardtype {sonet| sdh}**
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# hw-module subslot 0/3/0 sonet`
		Configures the controller for SONET framing.
		SONET framing (**sonet**) is the default. Whenever there is a change in framing mode (sonet/sdh), the SPA will be reloaded automatically. Reload will happen only when all the CEM Interface, T1 Controller and Sonet Controller configurations are removed completely. This is not applicable when you configure the first time because T1 controller and interface configurations would not exist.
		This configuration is mandatory for CEoP SPA to work normally in one of the framing modes. When you configure for the first time, it will not cause a SPA reload, if the cardtype is set to Sonet.

Step 3	**commit**	Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 4	**controller sonet** *interface-path-id*	Enters controller configuration submode and specifies the SONET controller name and instance identifier with the *rack/slot/module/port/controllerName* notation.
Step 5	**sts** *number* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# sts 1`	Configures the STS stream specified by *number*. The range is from1 to 3.
Step 6	**mode** *mode* **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# mode t1`	Sets the mode of interface at the STS level. The possible modes are: • vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1)
Step 7	**root** **Example:** `RP/0/RSP0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode. Go to step 7, if you want to create an structure agnostic CEM interface. Go to step 9, if you want to create a structure aware CEM interface.

	Command or Action	Purpose
Step 8	**controller t1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/0/1/0/1/4/1`	Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/sts-num/vtg-num/T1-num* notation.
Step 9	**cem-group unframed** **Example:** `RP/0/RSP0/CPU0:router(config)# cem-group unframed`	Creates an structure agnostic CEM interface.
Step 10	**controller t1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/0/1/0/1/5/1`	Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/sts-num/vtg-num/T1-num* notation.
Step 11	**cem-group framed** *group-number* **timeslots** *range1-range2* **Example:** `RP/0/RSP0/CPU0:router(config)# cem-group framed 0 timeslots 1`	Creates an structure aware CEM interface. The **timeslots** keyword specifies the time slots for the interface by range with the *range1-range2* notation.
Step 12	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

	Command or Action	Purpose

Step 13	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/0RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 14	**`show runn interface cem`** *`interface-path-id`*
	
	Example:
	`RP/0/RSP0/CPU0:router# show runn interface cem 0/0/2/0/1/1/1/1:1`
		Verifies the CEM interface configuration.

Configuring SDH AU-3 Mapped to C11-T1 or C12-E1

This section includes the following tasks:

- Configuring SDH AU-3 Mapped to C11-T1 and Creating CEM Interface, page 375
- Configuring SDH AU-3 Mapped to C12-E1 and Creating CEM Interface, page 378

Configuring SDH AU-3 Mapped to C11-T1 and Creating CEM Interface

This task explains how to configure SDH AU-3 with c11-t1 mapping.

Prerequisites

- You should know how to configure the SONET/SDH controller.

Restrictions

Channelized SDH AU-3 with c11-t1 mapping is supported on the following SPA:

- Cisco 1-Port Channelized OC-3/STM-1 SPA

SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*
3. **commit**
4. **controller sonet** *interface-path-id*
5. **au** *number*
6. **mode** *mode*
7. **root**
8. **controller t1** *interface-path-id*
9. **cem-group unframed**
10. **controller t1** *interface-path-id*
11. **cem-group framed** *group-number* **timeslots** *range1-range2*
12. **no shutdown**
13. **end**
    or
    commit
14. **show runn interface cem** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`hw-module sub-slot`** *`node-id`* **`cardtype`** *`type`*
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# hw-module sub-slot <> cardtype sdh`
		Configures the controller for Synchronous Digital Hierarchy (SDH) framing. The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in sonet/sdh mode.
		This command when committed results in automatic reload of SPA. Reload happens only when all the CEM interface, T1 Controller and Sonet Controller configurations are removed completely. This is not applicable when you configure the first time because T1 controller and interface configurations would not exist.
		This configuration is mandatory for CEoP SPA to work normally in one of the framing modes. SONET framing (**sonet**) is the default.

Step 3	`commit`	Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 4	**`controller sonet`** *`interface-path-id`*	Enters controller configuration submode and specifies the SDH controller name and instance identifier with the *rack/slot/module/port/controllerName* notation.

Step 5	**`au`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# au 1`
		Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is:
		• 1 to 3—1-Port Channelized OC-3/STM-1 SPA
		Note The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4.

Step 6	**`mode`** *`mode`*
	Example:
	`RP/0/RSP0/CPU0:router(config-auPath)# mode c11-t1`
		Sets the mode of interface at the AU level. AU-3 paths can be mapped to c11-t1 on supported SPAs.

Step 7	**`root`**
	Example:
	`RP/0/RSP0/CPU0:router(config-auPath)# root`
		Exits to global configuration mode.

Step 8	**`controller t1`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller T1 0/0/2/0/1/1/4`
		Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/t1Num* notation.

Step 9	**`cem-group unframed`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# cem-group unframed`
		Creates an structure agnostic CEM interface.

Step 10	**`controller t1`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t1 0/0/2/0/1/1/7`
		Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/t1Num* notation.

Step 11	**`cem-group framed`** *`group-number`* **`timeslots`** *`range1-range2`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# cem-group framed 1 timeslots 2-3`
		Creates an structure aware CEM interface. The **timeslots** keyword specifies the time slots for the interface by range with the *range1-range2* notation.

	Command or Action	Purpose

Step 12	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration.
		Note Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

Step 13	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 14	**`show runn interface cem`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show runn interface cem 0/0/2/0/1/1/1/1:1`
		Verifies the CEM interface configuration.

Configuring SDH AU-3 Mapped to C12-E1 and Creating CEM Interface

This task explains how to configure SDH AU-3 with c12-e1 mapping.

Prerequisites

- You should know how to configure the SONET/SDH controller.

Restrictions

Channelized SDH AU-3 with c12-e1 mapping is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA

SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*

3. **commit**
4. **controller sonet** *interface-path-id*
5. **au** *number*
6. **mode tug3**
7. **width** *number*
8. **tug3** *number*
9. **mode** *mode*
10. **root**
11. **controller e1** *interface-path-id*
12. **cem-group unframed**
13. **controller e1** *interface-path-id*
14. **cem-group framed** *group***-***number* **timeslots** *range1-range2*
15. **no shutdown**
16. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**hw-module sub-slot** *node-id* **cardtype** *type* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# hw-module sub-slot <> cardtype sdh`	Configures the controller framing for Synchronous Digital Hierarchy (SDH) framing. The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in sonet/sdh mode. This command when committed results in automatic reload of SPA. Reload happens only when all the CEM interface, E1 Controller and Sonet Controller configurations are removed completely.
Step 3	**commit**	Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 4	**controller sonet** *interface-path-id*	Enters controller configuration submode and specifies the SDH controller name and instance identifier with the *rack/slot/module/port/controllerName* notation.

	Command or Action	Purpose
Step 5	**au** *number* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# au 1`	Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is: • 1 to 3—1-Port Channelized OC-3/STM-1 SPA **Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4.
Step 6	**mode tug3** **Example:** `RP/0/RSP0/CPU0:router(config-auPath)# mode tug3`	Sets the mode of interface at the AU level. Currently only TUG3 is supported.
Step 7	**width** *number* **Example:** `RP/0/RSP0/CPU0:router(config-auPath)# width 3`	Configures the number of the AU streams.
Step 8	**tug3** *number* **Example:** `RP/0/RSP0/CPU0:router(config-auPath)#tug3 1`	Specifies the Tributary Unit Group (TUG) *number* and enters the config-tug3Path mode. The range is 1 to 3.
Step 9	**mode** *mode* **Example:** `RP/0/RSP0/CPU0:router(config-tug3Path)# mode c12-e1`	Sets the mode of interface at the tug3 level. The modes are: • **c12-e1**—TUG-3 path carrying TU-12 to E1
Step 10	**root** **Example:** `RP/0/RSP0/CPU0:router(config-auPath)# root`	Exits to global configuration mode.
Step 11	**controller e1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller E1 0/0/2/0/1/1/1`	Enters E1 controller configuration submode and specifies the E1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/tugNum/t1Num* notation.
Step 12	**cem-group unframed** **Example:** `RP/0/RSP0/CPU0:router(config)# cem-group unframed`	Creates an structure agnostic CEM interface.

	Command or Action	Purpose

Step 13	**`controller e1`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller E1 0/0/2/0/1/1/7`
		Enters E1 controller configuration submode and specifies the E1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/tugNum/t1Num* notation.

Step 14	**`cem-group framed`** *`group-number`* **`timeslots`** *`range1-range2`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# cem-group framed 0 timeslots 1`
		Creates an structure aware CEM interface.

Step 15	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration.
		Note Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).

Step 16	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring CEM Interface

This section provides information about how to configure CEM. CEM provides a bridge between a time-division multiplexing (TDM) network and a packet network using Multiprotocol Label Switching (MPLS). The router encapsulates the TDM data in the MPLS packets and sends the data over a CEM pseudowire to the remote provider edge (PE) router.

The following sections describe how to configure CEM:

- Configuration Guidelines and Restrictions

- Configuring a Global CEM Class
- Attaching a CEM Class
- Configuring Payload Size
- Setting the Dejitter Buffer Size
- Setting an Idle Pattern
- Enabling Dummy Mode
- Setting a Dummy Pattern

Configuration Guidelines and Restrictions

Not all combinations of payload size and dejitter buffer size are supported. If you apply an incompatible payload size or dejitter buffer configuration, the router rejects it and reverts to the previous configuration.

Configuring a Global CEM Class

This task explains how to configure a global CEM class.

Note Any interface configuration would have higher precedence over configuration applied through attaching a CEM class. Also, CEM class attached to an interface would have higher precedence than CEM class attached to the parent controller. For example, if the dummy pattern value of *0xcf* is applied directly to an interface and then a CEM class which contains dummy pattern value of *0xaa* is attached to the same interface, then the dummy pattern value would be *0xcf*. The new configuration would not be applied until the dummy pattern value applied directly to the interface is removed.

SUMMARY STEPS

1. **configure**
2. **cem class** *class-name*
3. **payload** *value*
4. **dejitter** *value*
5. **idle pattern** *value*
6. **dummy mode {last-frame|user-defined}**
7. **dummy pattern** *value*
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`cem class`** *`class-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# cem class Default`
		Creates a new CEM class.

Step 3	**`payload`** *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# payload 512`
		Enter the payload size for the CEM class.

Step 4	**`dejitter`** *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# dejitter 10`
		Enter the dejitter buffer size for the CEM class.

Step 5	**`idle pattern`** *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# idle pattern 0x55`
		Enter the idle pattern value for the CEM class.

Step 6	**`dummy mode`**
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# dummy mode last-frame`
		Enter the dummy mode for the CEM class. The options are last-frame or user-defined.

	Command or Action	Purpose

Step 7	**`dummy pattern`** *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# dummy pattern`
		Enter the dummy pattern value for the CEM class. This value is applied only when the dummy mode is user-defined.

Step 8	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# end`
	or
	`RP/0/RSP0/CPU0:router(config-cem-class)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Attaching a CEM Class

This task explains how to attach a global CEM class.

Note You can attach a CEM class either to a CEM interface or to a T1/E1 controller.

SUMMARY STEPS

1. **configure**
2. **interface cem** *`interface-path-id`* **(or) controller {t1|e1}** *rack/slot/subslot/port*
3. **cem class-attach** *class-name*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface cem** *interface-path-id* (or) **controller {t1|e1}** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/0/2/0/1/1`	Specifies the CEM interface or the T1/E1 controller.
Step 3	**cem class-attach** *class-name* **Example:** `RP/0/RSP0/CPU0:router(config)# cem class-attach Default`	Attaches the CEM class to an interface or controller.

Step 4	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-cem-class)# end`
	or
	`RP/0/RSP0/CPU0:router(config-cem-class)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Payload Size

To specify the number of bytes encapsulated into a single IP packet, use the **cem payload** command. The size argument specifies the number of bytes in the payload of each packet. The range is from 32 to 1312 bytes.

Default payload sizes for an unstructured CEM channel are as follows:

- E1 = 256 bytes
- T1 = 192 bytes
- E3 = 1024 bytes
- T3 = 1024 bytes

Default payload sizes for a structured CEM channel depend on the number of time slots that constitute the channel. Payload (L in bytes), number of time slots (N), and packetization delay (D in milliseconds) have the following relationship: L = 8*N*D.

The default payload size is calculated using the packetization latency depending on the number of time slots the cem interface represents. The relationship between the number of time slots and the packetization latency is provided below:

- For N = 1, D is 8 milliseconds (with the corresponding packet payloadsize of 64 bytes)
- For 2 <=N <= 4, D is 4 milliseconds (with the corresponding packetpayload size of 32*N bytes)
- For N >= 5, D is 1 millisecond (with the corresponding packet payloadsize of 8*N octets).

Support of 5 ms packetization latency for N = 1 is recommended.

Setting the Dejitter Buffer Size

To specify the size of the dejitter buffer used to compensate for the network filter, use the **cem dejitter** command. The configured dejitter buffer size is converted from milliseconds to packets and rounded up to the next integral number of packets. Use the size argument to specify the size of the buffer, in milliseconds. The range is from 1 to 500 ms. The following is an example:

```
Router(config-cem)# cem dejitter 5
```

The default dejitter buffer for a CEM channel, irrespective of CESoPSN or SAToP, is as follows:

- E1 = 16 milliseconds
- T1 = 16 milliseconds
- E3 = 5 milliseconds
- T3 = 5 milliseconds

Note Refer Table 6, Table 7, and Table 8 for the relationship between payload and dejitter buffer on SAToP T1/E1, T3/E3, and CESoPSN lines. Configuration of payload and dejitter should be in accordance with the minimum and maximum values as mentioned in the table.

Note The maximum and minimum dejitter buffer value, that is the range is fixed for a given payload value.

Setting an Idle Pattern

To specify an idle pattern, use the [no] **cem idle pattern pattern** command. The payload of each lost CESoPSN data packet must be replaced with the equivalent amount of the replacement data. The range for pattern is from 0x0 to 0xff; the default idle pattern is 0xff. This is an example:

```
Router(config-cem)# cem idle pattern 0xff
```

If the expected CEM packets are not received for a given CEM interface and are considered as being lost, then the CEoP SPA will play out the idle pattern towards the TDM attachment circuit in the respective timeslots configured in the CEM group.

Enabling Dummy Mode

Dummy mode enables a bit pattern for filling in for lost or corrupted frames. To enable dummy mode, use the **cem dummy mode** [**last-frame** | **user-defined**] command. The default is **last-frame**. This is an example:

```
Router(config-cem)# cem dummy mode last-frame
```

When packets are lost due to misordering or where reordering of packets is not successful, the CEoP SPA will play out the Dummy pattern towards the TDM attachment circuit in respective timeslots configured in the CEM group.

Setting a Dummy Pattern

If dummy mode is set to user-defined, you can use the **cem dummy-pattern** command to configure the dummy pattern. The range for pattern is from 0x0 to 0xff. The default dummy pattern is 0xff. This is an example:

```
Router(config-cem)# cem dummy-pattern 0xff
```

The Table 6 shows the relationship between payload and dejitter for T1/E1 SAToP lines.

Table 6 T1/E1 SAToP lines: Payload and Jitter Limits

T1/E1	Maximum Payload	Maximum Jitter	Minimum Jitter	Minimum Payload	Maximum Jitter	Minimun Jitter
T1	960	320	10	192	64	2
E1	1280	320	10	256	64	2

The Table 7 shows the relationship between payload and dejitter for T3/E3 SAToP lines.

Table 7 T3/E3 SAToP lines: Payload and Jitter Limits

T3/E3	Maximum Payload	Maximum Jitter	Minimum Jitter	Minimum Payload	Maximum Jitter	Minimun Jitter
T3	1312	8	2	672	8	2
E3	1312	16	2	512	8	2

The Table 8 shows the relationship between payload and dejitter for DS0 lines.

Table 8 CESoPSN DS0 Lines: Payload and Jitter Limits

DS0	Maximum Payload	Maximum Jitter	Minimum Jitter	Minimum Payload	Maximum Jitter	Minimun Jitter
1	40	320	10	32	256	8
2	80	320	10	32	128	4
3	120	320	10	33	128	4
4	160	320	10	32	64	2
5	200	320	10	40	64	2
6	240	320	10	48	64	2
7	280	320	10	56	64	2
8	320	320	10	64	64	2
9	360	320	10	72	64	2
10	400	320	10	80	64	2
11	440	320	10	88	64	2
12	480	320	10	96	64	2
13	520	320	10	104	64	2
14	560	320	10	112	64	2
15	600	320	10	120	64	2
16	640	320	10	128	64	2
17	680	320	10	136	64	2
18	720	320	10	144	64	2
19	760	320	10	152	64	2
20	800	320	10	160	64	2
21	840	320	10	168	64	2
22	880	320	10	176	64	2
23	920	320	10	184	64	2
24	960	320	10	192	64	2
25	1000	320	10	200	64	2
26	1040	320	10	208	64	2
27	1080	320	10	216	64	2
28	1120	320	10	224	64	2
29	1160	320	10	232	64	2
30	1200	320	10	240	64	2
31	1240	320	10	248	64	2
32	1280	320	10	256	64	2

Configuring Clocking

Each SPA port shall be configured either to use system clock from the host card or loop timed independently. Each SPA also supplies a reference clock to the host which can be selected among the received port clocks. This section provides information about how to configure clocking on the 1xOC3 SPA.

This section describes the following topics:

- Configuring Clock Recovery
- Verifying Clock recovery

Configuring Clock Recovery

When configuring clock recovery, consider the following guidelines:

Adaptive Clock Recovery

- Clock source:
  - In Cisco IOS XR Release 4.2.0 and later, recovered clock from a CEM interface on the 1-Port Channelized OC-3/STM1 CEoP SPA can be used as a clock source on the SPA itself.
- Number of clock sources allowed:
  - Refer the section Clock Distribution, page 370 for more information.
- The clock must be the same as used by the router as the network clock. Any pseudowire in this case can carry the clock.
- The minimum bundle size of CEM pseudowires on the network that delivers robust clock recovery is 4 DS0s.
- The minimum packet size of CEM pseudowires on the network that delivers robust clock recovery is 64 bytes.

Differential Clocking

- The maximum number of differential clocks sourced from a 1-Port Channelized OC-3/STM1 CEoP SPA is 10.
- The 1-Port Channelized OC-3/STM1 CEoP SPA can recover up to 10 T1/E1 clocks.
- There are several bundles sent from the same port. The bundle that is used for carrying the clock of the port is the first created bundle of the port. Only pseudowires that include the first DS0 of a port can carry differential clock.
- You must have a Stratum-1 clock, a common clock going to both PE routers. If not, the recovery will not work as expected.

To configure clock recovery on the CEoP SPA and to apply the recovered clock to the controller, use the following procedure:

SUMMARY STEPS

1. **configure**
2. **interface cem** *rack/slot/subslot/port:cem-group*
3. **transmit-clock** differential

4. **recover-clock** *clock-id* **{adaptive |differential}**
5. **controller {t1|e1|t3|e3}** *rack/slot/subslot/port*
6. **clock source recovered** *clock-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface cem`** `rack/slot/subslot/port:cem-group`
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface cem 0/1/0/0:2`
		Specifies the complete CEM interface instance.

Step 3	**`transmit-clock`** `{`**`differential`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# transmit-clock source internal`
		Configures the CEM port transmit clock source. This is typically configured at the node acting as Master to send the clock. This command is not required for Adaptive Clock Recovery.

Step 4	**`recover-clock`** *`clock-id`* **`{adaptive	differential}`** **Example:** `RP/0/RSP0/CPU0:router(config-if)# recover-clock clock-id <> adaptive`	Specifies the recovered clock number and the clock recovery type. This is typically configured at the node acting as Slave that recovers the clock from incoming CEM packets from core.

Step 5	**`controller`** *`name instance`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t1 0/1/1/0/0/0`
		Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/name/instance1/instance2* notation.

Step 6	**`clock source recovered`** *`clock-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# clock source recovered 3`
		Specifies the recovered clock number. This applies the recovered clock from a CEM interface on a T1/E1 Controller.

Verifying Clock recovery

To verify clock recovery, use the **show recovered-clock** command.

```
Router# show recovered-clock sublsot 0/3/0
Recovered clock status for subslot 0/3/0
----------------------------------------
Clock    Mode         Port CEM  Status     Frequency Offset(ppb)
1        ADAPTIVE     0    1    HOLDOVER   0
Router# show recovered-clock
Recovered clock status for subslot 3/0
----------------------------------------
Clock    Mode         Port CEM  Status     Frequency Offset(ppb)
1        ADAPTIVE     0    1    ACQUIRING  -694
```

Configuration Examples for CEM

This section contains the following examples:

Circuit Emulation Interface Configuration: Examples

The following example shows a sample CEM interface configuration on the Cisco 1-port Channelized OC3/STM-1 SPA.

Channelized Sonet / SDH Configurations and CEM Interface Creation

Sonet - T1 Channelization and CEM Interface Creation

```
hw-module subslot  <loc> cardtype sonet
controller SONET 0/0/1/0
 sts 1
  mode vt15-t1
sts 2
  mode vt15-t1
sts 3
  mode vt15-t1
commit
```

In case of structure agnostic cem interface:

```
controller T1 0/0/1/0/1/4/1
 cem-group unframed
```

In case of structure aware cem interface:

```
controller T1 0/0/1/0/1/5/1
 cem-group framed 0 timeslots 1
 cem-group framed 1 timeslots 2-3
 cem-group framed 2 timeslots 4-6
 cem-group framed 3 timeslots 7-10
 cem-group framed 4 timeslots 11-15
 cem-group framed 5 timeslots 16-21
 cem-group framed 6 timeslots 22-24
```

SDH - T1 Channelization and CEM Interface Creation

```
hw-module subslot  <loc> cardtype sdh
controller SONET0/0/2/0
 au 1
  mode c11-t1
 au 2
  mode c11-t1
 au 3
  mode c11-t1
commit
```

In case of structure agnostic cem interface:

```
controller T1 0/0/2/0/1/1/4
 cem-group unframed
```

In case of structure aware cem interface:

```
controller T1 0/0/2/0/1/7/1
 cem-group framed 0 timeslots 1
 cem-group framed 1 timeslots 2-3
 cem-group framed 2 timeslots 4-6
 cem-group framed 3 timeslots 7-10
 cem-group framed 4 timeslots 11-15
 cem-group framed 5 timeslots 16-21
 cem-group framed 6 timeslots 22-24
```

SDH - E1 Channelization and CEM Interface Creation

```
hw-module subslot  <loc> cardtype sdh
controller SONET 0/0/2/0
 au 1
  mode tug3
  width 3
  tug3 1
   mode c12-e1
tug3 2
   mode c12-e1
 tug3 3
   mode c12-e1
commit
```

In case of structure agnostic cem interface:

```
controller E1 0/0/2/0/1/1/1/1
 cem-group unframed
```

In case of structure aware cem interface:

```
controller E1 0/0/2/0/1/1/7/1
 cem-group framed 0 timeslots 1
 cem-group framed 1 timeslots 2-3
 cem-group framed 2 timeslots 4-6
 cem-group framed 3 timeslots 7-10
 cem-group framed 4 timeslots 11-15
 cem-group framed 5 timeslots 16-21
 cem-group framed 6 timeslots 22-31
```

CEM Interface Configuration

```
RP/0/RSP0/CPU0:CEOP-01#show runn interface cem 0/0/2/0/1/1/1/1:1

interface CEM0/0/2/0/1/1/1/1:1
 l2transport
 !
CEM Interface Config Options :

RP/0/RSP0/CPU0:CEOP-01(config)#interface cem 0/0/2/0/1/1/1/1:1
RP/0/RSP0/CPU0:CEOP-01(config-if)#cem ?
```

```
class-attach  Attach a CEM class to this interface
clock         Configure clocks on this CEM interface
dejitter      Configure dejitter buffer
dummy         Configure dummy frame parameters
idle          Configure idle frame parameters
payload       Configure payload size of CEM frames
```

Clock Recovery : Example

Adaptive Clock Recovery Configuration:

(E1 configurations are similar to T1s given below)

```
CE1
----
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source internal

PE1 (Acts as source of clock, but no specific configuration under CEM Interface is needed
here)
---------------------------------------------------------------------------------------
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source line

PE2 (On PE node where clock recovery is done):
----------------------------------------
```

To recover the adaptive clock:

```
Router(config)# interface cem 0/0/2/0/1/1/4:0
Router(config-if)#cem clock recover <clock-id> adaptive
```

To apply the recovered clock,

```
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source recovered <clock-id>

CE2
----
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source line
```

Differential Clock Recovery Configuration:

```
CE1
----
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source internal

PE1 (Acts as source of clock)
-----------------------------
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source line
Router(config)# interface cem 0/0/2/0/1/1/4:0
Router(config-if)#cem clock transmit differential

PE2 (To recover the differential clock):
---------------------------------------
Router (config)#interface cem 0/0/2/0/1/1/4:0
Router (config-t1)#cem clock recover <clock-id> differential
```

To apply the recovered clock:

```
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#cem clock recovered <clock-id>

CE2
----
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source line
```

Additional References

The following sections provide references to related documents.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco ASR 9000 Series Aggregation Services RouterInterface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using the Cisco IOS XR software	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
• CISCO-SONET-MIB • ENTITY-MIB • SONET-MIB (RFC 3592) The following additional MIBs are supported on the Cisco 1-Port Channelized OC-3/STM-1 SPA on the Cisco ASR 9000 Series Router: • CISCO-IF-EXTENSION-MIB • DS1-MIB • DS3-MIB • IF-MIB	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 5086, RFC 4553. RFC 4197, RFC 5287	• *Structure-Aware Time Division Multiplexed (TDM) Circuit Emulation Service over Packet Switched Network (CESoPSN)* • *Structure-Agnostic Time Division Multiplexing (TDM) over Packet (SAToP)* • *Requirements for Edge-to-Edge Emulation of Time Division Multiplexed (TDM) Circuits over Packet Switching Networks* • *Control Protocol Extensions for the Setup of Time-Division Multiplexing (TDM) Pseudowires in MPLS Networks*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router

This module describes the configuration of clear channel SONET controllers on the Cisco ASR 9000 Series Router.

SONET controller configuration is a prerequisite for configuring Inter-Chassis Stateful Switchover (ICSSO) for Point-to-Point Protocol (PPP) and Multilink PPP (MLPPP), channelized SONET, or serial interfaces on the Cisco ASR 9000 Series Router.

SONET allows you to define optical signals and a synchronous frame structure for multiplexed digital traffic. It is a set of standards defining the rates and formats for optical networks specified in American National Standards Institute (ANSI) T1.105, ANSI T1.106, and ANSI T1.117.

For more information about configuring a channelized SONET controller, see the "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" module.

The commands for configuring the Layer 1 SONET controllers are provided in the *Cisco IOS XR Interface and Hardware Component Command Reference*.

Feature History for Configuring SONET Controllers on Cisco IOS XR Software

Release	Modification
Release 3.9.0	Support for the following SPA was introduced on the Cisco ASR 9000 Series Router: • Cisco 2-Port Channelized OC-12c/DS0 SPA
Release 4.0.0	Support for the following SPAs was introduced on the Cisco ASR 9000 Series Router: • Cisco 1-Port Channelized OC-48/STM-16 SPA • Cisco 1-Port OC-192c/STM-64 POS/RPR XFP SPA • Cisco 2-Port OC-48c/STM-16 POS/RPR SPA • Cisco 8-Port OC-12c/STM-4 POS SPA
Release 4.0.1	Support for the following SPAs was introduced on the Cisco ASR 9000 Series Router: • Cisco 4-Port OC-3c/STM-1 POS SPA • Cisco 8-Port OC-3c/STM-1 POS SPA

Contents

Prerequisites for Configuring Clear Channel SONET Controllers

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring SONET controllers, be sure that the following tasks and conditions are met:

- You have one of the following SPAs installed:
  - Cisco 2-Port Channelized OC-12c/DS0 SPA
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
  - Cisco 4-Port OC-3c/STM-1 POS SPA
  - Cisco 8-Port OC-3c/STM-1 POS SPA
  - Cisco 1-Port OC-192c/STM-64 POS/RPR XFP SPA
  - Cisco 2-Port OC-48c/STM-16 POS/RPR SPA
  - Cisco 8-Port OC-12c/STM-4 POS SPA
- You know how to apply the specify the SONET controller name and instance identifier with the generalized notation *rack/slot/module/port*. The SONET controller name and instance identifier are required with the **controller sonet** command.

Information About Configuring SONET Controllers

To configure SONET controllers, you must understand the following concepts:

- SONET Controller Overview, page 398
- Default Configuration Values for SONET Controllers, page 399
- SONET APS, page 400

SONET Controller Overview

In routers supporting Cisco IOS XR software, the physical ports on certain line cards are called controllers. Before you can configure channelized SONET or a serial interface, you need to configure the SONET controller.

The commands used to configure the physical SONET port are grouped under the SONET controller configuration mode. To get to the SONET controller configuration mode, enter the **controller sonet** command in global configuration mode. You can also preconfigure a SONET controller using the **controller preconfigure sonet** global configuration command.

The router uses SONET controllers for Layer 1 and Layer 2 processing.

Default Configuration Values for SONET Controllers

Table 9 describes some default configuration parameters that are present on SONET controllers.

Table 9 SONET Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Reporting of the following alarms for a SONET controller: • Bit 1 (B1) bit error rate (BER) threshold crossing alert (TCA) errors • Bit 2 (B2) BER TCA errors • Signal failure BER errors • Section loss of frame (SLOF) errors • Section loss of signal (SLOS) errors	**enabled**	To disable reporting of any alarms enabled by default, use the **no report** [**b1-tca** | **b2-tca** | **sf-ber** | **slof** | **slos**] command in SONET/SDH configuration mode. To enable reporting of line alarm indication signal (LAIS), line remote defect indication (LRDI), or signal degradation BER errors, use the **report** [**lais** | **lrdi** | **sd-ber**] command in SONET/SDH configuration mode.
Reporting of the following alarms for a SONET path controller: • Bit 3 (B3) BER TCA errors • Path loss of pointer (PLOP) errors	**enabled**	To disable B3 BER TCA or PLOP reporting on the SONET path controller, enter the **no report b3-tca** or **no report plop** command in SONET/SDH path configuration submode. To enable reporting of path alarm indication signal (PAIS), path payload mismatch (PPLM), path remote defect indication (PRDI), or path trace identity mismatch (PTIM) errors, use the **report** [**pais** | **pplm** | **prdi** | **ptim** command in SONET/SDH path configuration submode.

Table 9 SONET Controller Default Configuration Values (continued)

Parameter	Default Value	Configuration File Entry
Synchronous payload envelope (SPE) scrambling	**enabled**	To disable SPE scrambling on a SONET controller, enter the **path scrambling disable** command in SONET controller configuration submode.
Keepalive timer	**enabled**	To turn off the keepalive timer, enter the **keepalive disable** command in interface configuration mode.

SONET APS

The automatic protection switching (APS) feature allows switchover of interfaces in the event of failure, and is often required when connecting SONET equipment to telco equipment. APS refers to the mechanism of using a *protect* interface in the SONET network as the backup for *working* interface. When the working interface fails, the protect interface quickly assumes its traffic load. The working interfaces and their protect interfaces make up an *APS group*.

In Cisco IOS XR software, SONET APS configuration defines a working line and a protection line for each redundant line pair. The working line is the primary or preferred line, and communications take place over that line as long as the line remains operative. If a failure occurs on the working line, APS initiates a switchover to the protection line. For proper APS operation between two routers, a working line on one router must also be the working line on the other router, and the same applies to the protection line.

In a SONET APS group, each connection may be bidirectional or unidirectional, and revertive or non-revertive. The same signal payload is sent to the working and protect interfaces. The working and protect interfaces terminate in two different routers.

The protect interface directs the working interface to activate or deactivate in the case of degradation, loss of channel signal, or manual intervention. If communication between the working and protect interfaces is lost, the working router assumes full control of the working interface as if no protect circuit existed.

In an APS group, each line is called a *channel*. In bidirectional mode, the receive and transmit channels are switched as a pair. In unidirectional mode, the transmit and receive channels are switched independently. For example, in bidirectional mode, if the receive channel on the working interface has a loss of channel signal, both the receive and transmit channels are switched.

How to Configure Clear Channel SONET Controllers

This section contains the following procedures:

- Configuring a Clear Channel SONET Controller, page 401
- Configuring SONET APS, page 404
- Configuring a Hold-off Timer to Prevent Fast Reroute from Being Triggered, page 409

Configuring a Clear Channel SONET Controller

This task explains how to configure SONET controllers as a prerequisite to configuring POS or serial interfaces.

Prerequisites

- You need to have a supported POS SPA or channelized SPA installed in a router that is running the corresponding supported Cisco IOS XR software release.
- If you want to ensure recovery from fiber or equipment failures, then configure SONET APS on the router as describe in the "Configuring SONET APS" section on page -404.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **line delay trigger** *value*
5. **line delay clear** *value*
6. **framing** {**sdh** | **sonet**}
7. **loopback** {**internal** | **line**}
8. **overhead** {**j0** | **s1s0**} *byte-value*
9. **path** *keyword* [*values*]
10. **end**
    or
    commit
11. **show controllers sonet** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`controller sonet` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/0/0`
		Enters SONET controller configuration submode and specifies the SONET controller name and instance identifier with the *rack/slot/module/port* notation.

	Command or Action	Purpose

Step 3	**clock source** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back-to-back or over fiber for which no clocking is available.
		Note The **line** clock is the default.

Step 4	**line delay trigger** *value* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# line delay trigger 3000`	(Optional) Configures the SONET line delay trigger values, where the trigger values are in the range from 0 through 60000 milliseconds, and the default delay trigger value is 0 milliseconds.
Step 5	**line delay clear** *value* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# line delay clear 4000`	(Optional) Configures the amount of time before a SONET line delay trigger alarm is cleared. The range is from 1000 through 180000 milliseconds, and the default is 10 seconds.

Step 6	**framing** {**sdh** | **sonet**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# framing sonet`
		(Optional) Configures the controller framing with either the **sdh** keyword for Synchronous Digital Hierarchy (SDH) framing or the **sonet** keyword for SONET framing.
		SONET framing (**sonet**) is the default.

Step 7	**loopback** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# loopback internal`
		(Optional) Configures the SONET controller for loopback, where the **internal** keyword selects internal (terminal) loopback, or the **line** keyword selects line (facility) loopback.

Step 8	**overhead** {**j0** | **s1s0**} *byte-value* **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# overhead s1s0`	(Optional) Configures the controller's overhead, where the **j0** keyword specifies the STS identifier (J0/C1) byte, and the **s1s0** keyword specifies bits s1 and s0 of H1 byte. • The default byte value for the **j0** keyword is 0xcc, and the default byte value for the **s1s0** keyword is 0. • The range of valid values for **j0** and **s1s0** is 0 through 255.

	Command or Action	Purpose
Step 9	`path` *keyword* [*values*] **Example:** `RP/0/RSP0/CPU0:router(config-sonet)# path delay trigger 25`	(Optional) Configures SONET controller path values. Keyword definitions are as follows: • **ais-shut**—Set sending path alarm indication signal (PAIS) when shut down. • **b3-ber-prdi**—Enable sending of a path-level remote defect indication (PRDI) when the bit error rate (BER) bit interleaved parity (BIP) threshold is exceeded. • **delay clear** *value*—Set the amount of time before a Synchronous Transport Signal (STS) path delay trigger alarm is cleared. Replace the *value* argument with a number in the range from 0 through 180000 milliseconds. The default value is 10 seconds. • **delay trigger** *value* —Set SONET path delay values or delay trigger value. Replace the *value* argument with a number in the range from 0 through 60000 milliseconds. The default value is 0 milliseconds. • **overhead** [**c2** *byte-value* | **j1** *line*] —Set SONET POH byte or bit values. Enter the **c2** keyword to specify STS SPE content (C2) byte, and replace the *byte-value* argument with a number in the range from 0 through 255. Enter the **j1** keyword to configure the SONET path trace (J1) buffer, and replace the *line* argument with the path trace buffer identifier (in ASCII text). • **report [b3-tca | pais | plop | pplm | prdi | ptim]**—Set SONET path alarm reporting. Specifies which alarms are reported and which bit error rate (BER) thresholds will signal an alarm. By default, B3 BER threshold crossing alert (TCA) and path loss of pointer (PLOP) reporting are enabled. Specifying the **pais** keyword sets PAIS reporting status; **pplm** sets path payload mismatch (PPLM) defect reporting status; **prdi** sets path remote defect indication reporting status; and **ptim** sets path trace identity mismatch (PTIM) defect reporting status. The **no report b3-tca** and **no report plop** commands in SONET/SDH path configuration submode disable B3 BER TCA and PLOP reporting status, respectively. • **scrambling disable**—Disable SPE scrambling. Note that SPE scrambling is enabled by default. • **threshold b3-tca** *BER*—Set SONET path BER threshold value. Replace the *BER* argument with a number in the range from 3 through 9. The threshold value is interpreted as a negative exponent of 10 when determining the bit error rate. For example, a value of 5 implies a bit error rate of 10 to the minus 5. The default BER threshold value is 6. • **uneq-shut—**Sets sending Unequipped (UNEQ) when shut down.

	Command or Action	Purpose

Step 10	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 11	**`show controllers sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show controllers sonet 0/1/0/0`
		Verifies the SONET controller configuration.

Configuring SONET APS

SONET APS offers recovery from fiber (external) or equipment (interface and internal) failures at the SONET line layer. This task explains how to configure basic automatic protection switching (APS) on the router and how to configure more than one protect or working interface on a router by using the **aps group** command.

To verify the configuration or to determine if a switchover has occurred, use the **show aps** command.

Prerequisites

Before you configure SONET APS, be sure that you have a supported channelized SPA installed in a router that is running Cisco IOS XR software.

On the Cisco ASR 9000 Series Router, you must have a 2-Port Channelized OC-12c/DS0 SPA installed.

Restrictions

Before you configure SONET APS, consider the following restictions:

- The POS SPAs on the Cisco ASR 9000 Series Router do not support either single router or multirouter APS.

- The Cisco ASR 9000 Series Router supports multirouter APS on the 2-Port Channelized OC-12/DS0 SPA.
- For proper APS operation between two routers, a working line on one router must also be the working line on the other router, and the same applies to the protection line.

SUMMARY STEPS

1. **configure**
2. **aps group** *number*
3. **channel** {**0** | **1**} **local sonet** *interface*
4. Repeat Step 3 for each channel in the APS group.
5. **exit**
6. **interface loopback** *number*
7. **ipv4 address** *ip-address mask*
8. **exit**
9. **interface pos** *interface-path-id*
   or
   interface serial *interface-path-id*
10. **ipv4 address** *ip-address mask*
11. **pos crc** {**16** | **32**}
    or
    crc {**16** | **32**}
12. **encapsulation** {**frame-relay** | **hdlc** | **ppp**} (Serial interfaces only)
13. **keepalive** {*interval* | **disable**}[*retry*]
14. **no shutdown**
15. Repeat Step 9 to Step 13 for each channel in the group.
16. **exit**
17. **controller sonet** *interface-path-id*
18. **ais-shut**
19. **path scrambling disable**
20. **clock source** {**internal** | **line**}
21. Repeat Step 16 to Step 19 for each channel of the group.
22. **end**
    or
    commit
23. **exit**
24. **exit**
25. **show aps**
26. **show aps group** [*number*]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`aps group`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# aps group 1`
		Adds an APS group with a specified number and enters APS group configuration mode.
		- Use the **aps group** command in global configuration mode.
		- To remove a group, use the **no** form of this command, as in: **no aps group** *number,* where the value range is from 1–255.
		Note To use the **aps group** command, you must be a member of a user group associated with the proper task IDs for aps commands.
		Note The **aps group** command is used even when a single protect group is configured.

Step 3	**`channel`** `{`**`0`** `|` **`1`**`}` **`local sonet`** *`interface`*
	Example:
	`RP/0/RSP0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1`
		Creates a channel for the APS group. **0** designates a protect channel, and **1** designates a working channel.
		Note If the protect channel is local, it must be assigned using the **channel** command *before* any of the working channels is assigned.

Step 4	Repeat Step 3 for each channel in the group.	—
Step 5	**`exit`**	Exits APS group configuration mode and enters global configuration mode.

Step 6	**`interface loopback`** *`number`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface loopback 1`
		(Optional) Configures a loopback interface if a two-router APS is desired and enters interface configuration mode for a loopback interface.
		Note In this example, the loopback interface is used as the interconnect.

Step 7	**`ipv4 address`** *`ip-address mask`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.0.1 255.255.255.224`
		Assigns an IPV4 address and subnet mask to the loopback interface.

Step 8	**`exit`**	Exits interface configuration mode for a loopback interface, and enters global configuration mode.

	Command or Action	Purpose

Step 9	**`interface pos`** *`interface-path-id`*
	`or`
	`interface serial` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface POS 0/2/0/0`
	`or`
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0`
		Connects the interface for the channel selected in Step 3, and enters interface configuration mode.
		For serial interfaces, specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

Step 10	**`ipv4 address`** *`ip-address mask`*
	Example:
	`RP/0//CPU0:router(config-if)# ipv4 address 172.18.0.1 255.255.255.224`
		Assigns an IPv4 address and subnet mask to the interface.

Step 11	**`pos crc`** `{`**`16`** `|` **`32`**`}`
	`or`
	`crc` `{`**`16`** `|` **`32`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# pos crc 32`
	`or`
	`RP/0/RSP0/CPU0:router(config-if)# crc 32`
		Selects a CRC value for the channel. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode. For POS interfaces, the default CRC is 32. For serial interfaces, the default is 16.

Step 12	**`encapsulation`** `{`**`frame-relay`** `|` **`hdlc`** `|` **`ppp`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`
		(Serial interfaces only) Set the Layer 2 encapsulation of an interface.

Step 13	**`keepalive`** `{`*`interval`* `|` **`disable`**`}[`*`retry`*`]`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# keepalive disable`
		Sets the keepalive timer for the channel, where:
		• *interval*—Number of seconds (from 1 to 30) between keepalive messages. The default is 10.
		• **disable**—Turns off the keepalive timer.
		• *retry*—(Optional) Number of keepalive messages (from 1 to 255) that can be sent to a peer without a response before transitioning the link to the down state. The default is 5 for interfaces with PPP encapsulation, and 3 for interfaces with HDLC encapsulation.
		The **keepalive** command does not apply to interfaces using Frame Relay encapsulation.

Step 14	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# no shutdown`
		Removes the shutdown configuration.
		• The removal of the shutdown configuration removes the forced administrative down on the interface, enabling that interface to move to an up or down state (assuming the parent SONET layer is not configured administratively down).

Step 15	Repeat Step 9 through Step 13 for each channel in the group.	—

	Command or Action	Purpose
Step 16	**`exit`**	Exits interface configuration mode, and enters global configuration mode.

Step 17	**`controller sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/1/0/0`
		Enters SONET controller configuration mode and specifies the SONET controller name and instance identifier with the *rack/slot/module/port* notation.

Step 18	**`ais-shut`**
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# ais-shut`
		Configures SONET path values such as alarm indication signal (AIS) at shut down.

Step 19	**`path scrambling disable`**
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# path scrambling disable`
		(Optional) Disables synchronous payload envelope (SPE) scrambling.
		Note SPE scrambling is enabled by default.

Step 20	**`clock source`** {**`internal`** | **`line`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# clock source internal`
		Configures the SONET port TX clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.
		• Use the **line** keyword whenever clocking is derived from the network; use the **internal** keyword when two routers are connected back-to-back or over fiber for which no clocking is available.
		• The line clock (**line**) is the default.

Step 21	Repeat Step 16 through Step 19 for each channel in the group.	—

Step 22	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 23	**exit**	Exits SONET controller configuration mode, and enters global configuration mode.
Step 24	**exit**	Exits global configuration mode, and enters EXEC mode.
Step 25	**show aps** **Example:** `RP/0/RSP0/CPU0:router# show aps`	(Optional) Displays the operational status for all configured SONET APS groups.
Step 26	**show aps group** [*number*] **Example:** `RP/0/RSP0/CPU0:router# show aps group 3`	(Optional) Displays the operational status for configured SONET APS groups. **Note** The **show aps group** command is more useful than the **show aps** command when multiple groups are defined.

Configuring a Hold-off Timer to Prevent Fast Reroute from Being Triggered

When APS is configured on a router, it does not offer protection for tunnels; because of this limitation, fast reroute (FRR) still remains the protection mechanism for Multiprotocol Label Switching (MPLS) traffic-engineering.

When APS is configured in a SONET core network, an alarm might be generated toward a router downstream. If the router downstream is configured with FRR, you may want to configure a hold-off timer at the SONET level to prevent FRR from being triggered while the CORE network is doing a restoration. Perform this task to configure the delay.

Prerequisites

Configure SONET APS, as describe in the "Configuring SONET APS" section on page 404.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **line delay trigger** *value*
   or
   path delay trigger *value*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller sonet`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller sonet 0/6/0/0`
		Enters SONET configuration mode.

Step 3	**`line delay trigger`** *`value`*
	or
	`path delay trigger` *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# line delay trigger 250`
	`or`
	`RP/0/RSP0/CPU0:router(config-sonet)# path delay trigger 300`
		Configures SONET port delay trigger values in milliseconds.
		Tip The commands in Step 2 and Step 3 can be combined in one command string and entered from global configuration mode like this: **controller sonet** *r/s/m/p* **line delay trigger** or **controller sonet** *r/s/m/p* **path delay trigger**.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-sonet)# end`
	or
	`RP/0/RSP0/CPU0:router(config-sonet)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuration Examples for SONET Controllers

This section contains the following examples:

- SONET Controller Configuration: Example, page 411
- SONET APS Group Configuration: Example, page 411

SONET Controller Configuration: Example

The following example shows the commands and output generated when you are performing the configuration of a SONET controllers following the steps outlined in the "Configuring a Clear Channel SONET Controller" section on page 401. This example shows the usage of every optional command, along with listings of options within commands where relevant. An actual configuration may or may not include all these commands.

```
configure
controller sonet 0/1/0/0
    ais-shut
    clock source internal
    framing sonet
    loopback internal
Loopback is a traffic-effecting operation
    overhead s1s0 1
    path ais-shut
    path delay trigger 0
    path overhead j1 line 11
    path report pais
    path scrambling disable
    path threshold b3-tca 6
    path uneq-shut
    report pais
    threshold b2-tca 4
    commit
```

SONET APS Group Configuration: Example

The following example shows SONET Remote (two routers) APS configuration.

```
RP/0/0/CPU0:router(config)# aps group 1
    channel 0 local SONET 0/0/0/1
    channel 1 remote 172.18.69.123
    signalling sonet
    commit
show aps
show aps group 3
```

Additional References

The following sections provide references related to SONET controller configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using the Cisco IOS XR Software	*Cisco IOS XR Getting Started Guide*
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router

This module describes the configuration of clear channel T3/E3 controllers and channelized T3 and T1/E1 controllers on the Cisco ASR 9000 Series Aggregation Services Routers.

You must configure the T3/E3 controller before you can configure an associated serial interface.

Feature History for Configuring T3/E3 Controller Interfaces

Release	Modification
Release 3.9.0	This feature was introduced on the Cisco ASR 9000 Series Router for the Cisco 2-Port Channelized OC-12c/DS0 SPA.
Release 4.0.0	Support for the following features was added on the Cisco 2-Port Channelized OC-12c/DS0 SPA: • NxDS0 channelization • Link Noise Monitoring Support for clear channel T3 controllers on the 1-Port Channelized OC-48/STM-16 SPA was introduced.
Release 4.0.1	Support for the following SPAs was added: • Cisco 1-Port Channelized OC-3/STM-1 SPA • Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA
Release 4.1.0	• Support for the following SPAs was added: – Cisco 4-Port Channelized T3/DS0 SPA – Cisco 8-Port Channelized T1/E1 SPA • Support for a Link Noise Monitoring enhancement was added on the Cisco 2-Port Channelized OC-12c/DS0 SPA to set thresholds for noise errors on T1/E1 links that are used to signal the Noise Attribute to PPP for removal of an MLPPP bundle link.

Contents

Prerequisites for Configuring T3/E3 Controllers

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring T3/E3 controllers, be sure that you have one of the following supported SPAs installed in the router:

- Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA
- Cisco 4-Port Channelized T3/DS0 SPA

Note The 4-Port Channelized T3/DS0 SPA can run in clear channel mode, or it can be channelized into 28 T1 or 21 E1 controllers.

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 2-Port Channelized OC-12c/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 8-Port Channelized T1/E1 SPA
- Before you can configure a clear channel T3 controller on the channelized SONET SPAs, you must configure the SPA for an STS stream channelized for T3. For more information, see the "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" module.

Information About T3/E3 Controllers and Serial Interfaces

The 2-Port and 4-Port Clear Channel T3/E3 SPAs support clear channel services over serial lines only. The 4-Port Channelized T3/DS0 SPA supports clear channel services and channelized serial lines. If a controller is not channelized, then it is a clear channel controller, and the full bandwidth of its associated serial line is dedicated to a single channel that carries serial services.

When a T3 controller is channelized, it is logically divided into smaller bandwidth T1 or E1 controllers, depending on which mode of channelization you select. The sum of the bandwidth of the serial interfaces on the T1 or E1 controllers cannot exceed the bandwidth of the T3 controller that contains those channelized T1 or E1 controllers.

When you channelize a T3 controller, each individual T1 or E1 controller is automatically further channelized into DS0 time slots. A single T1 controller carries 24 DS0 time slots, and a single E1 controller carries 31 DS0 time slots. Users can divide these DS0 time slots up into individual channel groups. Each channel group can support a single serial interface.

When a controller is channelized, and channel groups have been created, services are provisioned on the associated serial interfaces.

The channelization feature in this release allows the following types of channelization:

- A single T3 controller into 28 T1 controllers, for a total controller size of 44210 kbps.
- A single T3 controller into 21 E1 controllers, for a total controller size of 43008 kbps.
- A single T1 controller supports up to 1.536 MB.
- A single E1 controller supports up to 2.048 MB.

Note A single shared port adapter (SPA) can support up to 448 channel groups.

This section includes the following additional topics:

- Supported Features, page 417
- Configuration Overview, page 422
- Default Configuration Values for T3 and E3 Controllers, page 422
- Default Configuration Values for T1 and E1 Controllers, page 423
- Link Noise Monitoring on T1 or E1 Links, page 424

Supported Features

Table 10 shows a summary of some of the supported features by SPA type.

Table 10 Supported Features on Channelized T3/E3, T1/E1, and Clear Channel SPAs

	1-Port Channelized OC-3/STM-1 SPA	**2-Port Channelized OC-12c/DS0 SPA**	**1-Port Channelized OC-48/STM-16 SPA**	**4-Port Channelized T3/DS0 SPA**	**8-Port Channelized T1/E1 SPA**	**2-Port and 4-Port Clear Channel T3/E3 SPA**
Bit Error Ratio Test (BERT)	T3, T1, E3, E1, and DS0 channels Maximum of 12 sessions[1] Maximum 1 session for T1	T3 channels	T3 and E3 Maximum of 2 simultaneous BERT tests are possible per STS-12.	T3, T1, E1 and DS0 channels	T1, E1, and DS0 channels	T3 and E3 1 session per port
Channelization and Clear Channel Modes	Channelized SONET/SDH Channelized T1/E1 to DS0s Clear channel SONET Clear channel T3 and E3 in SDH mode for serial interfaces	Channelized SONET/SDH Channelized T3/E3 Channelized T1/E1 to DS0s Clear channel SONET	Channelized SONET/SDH Channelized T3/E3 Clear channel SONET	Channelized T3 Channelized T1/E1 T3 clear channel	Channelized T1/E1 to DS0s. Clear channel T1 and E1	Clear channel T3 or E3 only

Table 10 Supported Features on Channelized T3/E3, T1/E1, and Clear Channel SPAs (continued)

	1-Port Channelized OC-3/STM-1 SPA	2-Port Channelized OC-12c/DS0 SPA	1-Port Channelized OC-48/STM-16 SPA	4-Port Channelized T3/DS0 SPA	8-Port Channelized T1/E1 SPA	2-Port and 4-Port Clear Channel T3/E3 SPA
DSU Modes	Adtran Digital-link Cisco Kentrox Larscom Verilink **For E3** Cisco (Default) Digital Link Kentrox	Adtran Digital-link Cisco Kentrox Larscom Verilink	Adtran Digital-link Cisco Kentrox Larscom Verilink **Note** Subrate for E3 is not supported.	Adtran Digital-link Cisco Kentrox Larscom Verilink	Adtran Digital-link Cisco Kentrox Larscom Verilink	Adtran Digital-link Cisco Kentrox Larscom Verilink
Encapsulations	Frame Relay HDLC PPP	HDLC PPP	Frame Relay HDLC PPP	Frame Relay HDLC PPP	Frame Relay HDLC PPP	Frame Relay HDLC PPP
Equal Cost Multipath (ECMP)	Yes	ECMP support for egress paths over T3 or T1 speed channels with either PPP or HDLC encapsulation ECMP support for paths on multiple controllers, SPAs, and SIPs	Yes	Yes	Yes	Yes
Facility Data Link (FDL)	Yes	Yes	Yes	Yes	Yes	No
Far End Alarm Control (FEAC)	For T3 C-bit framing	For T3 C-bit framing	For T3 C-bit framing	For T3 C-bit framing		For T3 C-bit framing
Inter-Chassis Stateful Switchover (ICSSO)[2]	For PPP on T3, T1, and E1 channels only (not DS0) For MLPPP on T1 and E1 sessions	For PPP on T3 channels For T1 when T3 channels are configured on the same system, SIP, SPA or port	No	T3, T1, and E1 channels only (not DS0)	T1 and E1 channels only (not DS0)	No

Table 10 Supported Features on Channelized T3/E3, T1/E1, and Clear Channel SPAs (continued)

	1-Port Channelized OC-3/STM-1 SPA	2-Port Channelized OC-12c/DS0 SPA	1-Port Channelized OC-48/STM-16 SPA	4-Port Channelized T3/DS0 SPA	8-Port Channelized T1/E1 SPA	2-Port and 4-Port Clear Channel T3/E3 SPA
IP Fast Reroute (IP-FRR)	No	For PPP only	No	T3, T1, and E1 channels	T1 and E1 channels	No
Link Noise Monitoring	No	Yes	No	No	No	No
Loopback[3]	Yes	Yes	Yes	Yes—Not DS0	Yes—Not DS0	Yes
Maintenance Data Link (MDL) Message Support	Yes	Yes	Yes	Yes	N/A	Yes
Circuit Emulation Service Over Packet Switched Network Support	Yes	Yes	No	No	No	No
Mixed Channel Support	No—T3 and E3 cannot be mixed. T1 and E1 cannot coexist on single STS-1.	Yes—T3 and T1 channels supported on the same SIP, SPA, or port	Yes	Yes	No—All channels must be either in T1 or E1 mode.	No—All ports must be either T3 or E3.
Scalability	1000 channels per SPA	48 T3 channels per SIP 24 T3 channels per SPA 12 T3 channels per interface	48 T3/E3 channels	1000channels per SPA	8 T1 or E1 ports Up to 256 full-duplex HDLC channels *N*x64K or *N*x56K channel speeds, where *N* is less than or equal to 24 for T1, and less than or equal to 32 for E1	2 or 4 T3 or E3 ports

[1] 6 simultaneous BERT sessions among first three physical ports and 6 simultaneous BERT sessions on 4th port.

[2] All interfaces configured on a SONET/SDH controller for the 1-Port Channelized OC-3/STM-1 SPA should be IC-SSO protected or none of them should be IC-SSO protected.

3. For detailed information about loopback support, see the "Loopback Support" section on page 420.

Loopback Support

Cisco 1-Port Channelized OC-3/STM-1 SPA

This section describes the types of loopback supported on the 1-Port Channelized OC-3/STM-1 SPA:

- For SONET controller:
  - Local loopback
  - Network line loopback
- For T3:
  - Local loopback
  - Network loopback
  - Remote loopback line (Use FEAC in C-Bit mode for T3)
  - Remote loopback payload (Use FEAC in C-Bit mode for T3)
- For E3:
  - Local loopback
  - Network loopback
- For T1:
  - Local loopback
  - Network line loopback
  - Remote line FDL ANSI loopback (also known as Remote CSU loopback - ESF mode)
  - Remote line FDL Bellcore loopback (also known as Remote SmartJack loopback - ESF mode)
  - Remote line inband loopback (SF inband loopback)
  - Remote payload FDL ANSI loopback (ESF remote payload loopback)
- For E1:
  - Local loopback
  - Network line loopback

Cisco 2-Port Channelized OC-12c/DS0 SPA

This section describes the types of loopback supported on the 2-Port Channelized OC-12c/DS0 SPA:

- For T3
  - Local loopback
  - Network line loopback
- For ports
  - Local line loopback
  - Network line loopback

Cisco 1-Port Channelized OC-48/STM-16 SPA

This section describes the types of loopback supported on the 1-Port Channelized OC-48/STM-16 SPA:

- For SONET:
  - Local line loopback
  - Network line loopback
- For T3:
  - Local loopback
  - Network line loopback
  - Network payload loopback
- For E3:
  - Local loopback
  - Network loopback

Cisco 4-Port Channelized T3/DS0 SPA

This section describes the types of loopback supported on the 4-Port Channelized T3/DS0 SPA:

- For T3:
  - Local loopback
  - Network loopback
  - Remote loopback line
- For T1:
  - Local loopback
  - Network line loopback
  - Remote line FDL ANSI loopback (also known as Remote CSU loopback - ESF mode)
  - Remote line FDL Bellcore loopback (also known as Remote SmartJack loopback - ESF mode)
- For E1:
  - Local loopback
  - Network line loopback

Cisco 8-Port Channelized T1/E1 SPA

This section describes the types of loopback supported on the 8-Port Channelized T1/E1 SPA:

- For T1:
  - Local loopback
  - Networkl line loopback
  - Remote line FDL ANSI loopback (also known as Remote CSU loopback - ESF mode)
  - Remote line FDL Bellcore loopback (also known as Remote SmartJack loopback - ESF mode)
- For E1:
  - Local loopback

Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA

This section describes the types of loopback supported on the 2-Port and 4-Port Clear Channel T3/E3 SPA:

- Local loopback
- Network payload loopback (Configure the local framer to send all data received from the remote side back to the remote side.)
- Network line loopback (Configure the local LIU to send all data received from the remote side back to the remote side.)
- Remote line loopback (Use FEAC to request the remote interface to loop back to SPA—T3 only)

Configuration Overview

Configuring a channelized T3 controller and its associated serial interfaces is a 4-step process:

Step 1 Configure the T3 controller, and set the mode for that controller to T1 or E1.

Step 2 Configure the T1 or E1 controller.

Step 3 Create channel groups and assign DS0 time slots to these channel groups as desired.

Step 4 Configure the serial interfaces that are associated with the individual channel groups, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module later in this document.

Default Configuration Values for T3 and E3 Controllers

Table 11 describes the default configuration parameters that are present on the T3 and E3 controllers.

Note

- Auto-detect framing is not supported on the 2-Port Channelized OC-12c/DS0 SPA.
- E3 is not supported on the 4-Port Channelized T3/DS0 SPA.

Table 11 T3 and E3 Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Frame type for the data line	For T3: C-bit framing For E3: G.751	**framing** {**auto-detect** | **c-bit** | **m23**}
Clocking for individual T3/E3 links	**internal**	**clock source** {**internal** | **line**}
Cable length	224 feet	**cablelength** *feet*

Table 11 T3 and E3 Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Maintenance data link (MDL) messages (T3 only)	**disable**	**mdl transmit** {**idle-signal** | **path** | **test-signal**} {**disable** | **enable**}
National reserved bits for an E3 port (E3 only)	**enable**, and the bit pattern value is 1.	**national bits** {**disable** | **enable**}

Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Default Configuration Values for T1 and E1 Controllers

Table 12 describes the default configuration parameters that are present on the T1 and E1 controllers.

Table 12 T1 and E1 Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Frame type for the data line	For T1: extended superframe (**esf**)For E1: framing with CRC-4 error monitoring capabilities (**crc4**).	For T1: **framing** {**sf** | **esf**}For E1: **framing** {**crc4** | **no-crc4** | **unframed**
Detection and generation of T1 yellow alarms. (T1 only)	Yellow alarms are detected and generated on the T1 channel.	**yellow** {**detection** | **generation**} {**disable** | **enable**}
Clocking for individual T1 and E1 links	**internal**	**clock source** {**internal** | **line**}
Cable length (T1 only)	For **cablelength long** command: *db-gain-value*: gain26; *db-loss-value*: 0db. For **cablelength short** command: 533 feet.	To set a cable length of longer than 655 feet: **cablelength long** *db-gain-value db-loss-value* To set a cable length of 655 feet or shorter: **cablelength short** *length*
Transmission of ANSI T1.403 or AT&T TR54016 once-per-second performance reports through Facility Data Link (FDL) for a T1 channel (T1 only)	**disable**	**fdl** {**ansi** | **att**} {**enable** | **disable**}
National reserved bits for an E1 port (E1 only)	0 (which corresponds to *0x1f* in hexadecimal format)	**national bits** *bits*

Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Link Noise Monitoring on T1 or E1 Links

Link Noise Monitoring (LNM) provides the ability to monitor Path Code Violation (PCV) errors on T1 and E1 links on the 2-Port Channelized OC-12c/DS0 SPA on the Cisco ASR 9000 Series Router, and to signal events and alarms on these links when noise continuously meets or exceeds configured thresholds (the **set** threshold values) for those errors. Events are also signaled when noise falls below configured improved thresholds (the **clear** threshold values).

Beginning in Cisco IOS XR Release 4.1, the LNM feature supports the **lnm remove** command to signal the Noise Attribute to PPP to remove an MLPPP bundle member link when specified thresholds are crossed.

Note An LCV is an occurrence of either a Bi-Polar Violation (BPV) or Excessive Zeroes (EXZ) error, and a PCV is an occurrence of a CRC error in a timeslot. However, the LNM feature currently only monitors PCV errors. The LCV values are only used to calculate an expected PCV if the PCV values are not specified. If the PCV values are specified, then the LCV values are ignored.

LNM Events

There are two basic types of monitoring events produced by LNM:

- Crossed events—A *crossed* event signals when PCV threshold values continuously meet or exceed the specified **set** values for major and minor warnings for a specified period of time (**duration**). When a crossed event occurs, the major or minor monitoring type for the controller is reported as the *alarm* state. When the crossed event is no longer present, the monitoring type returns to the *stable* state.

  The following are examples of crossed events:

```
RP/0/RSP0/CPU0:Router#0/1/CPU0:May 13 9:54:10.980 : g_spa_1[181]:
%L2-T1E1_LNM-3-MINWARNNOISE :
Interface T10/1/1/0/1/1/1, noise crossed minor warning threshold

RP/0/RSP0/CPU0:Router#0/1/CPU0:May 13 9:54:11.980 : g_spa_1[181]:
%L2-T1E1_LNM-3-MAJWARNNOISE :
Interface T10/1/1/0/1/1/1, noise crossed major warning threshold
```

- Cleared events—A *cleared* event signals when threshold values that were crossed have fallen below the specified **clear** values for major and minor warnings.

  The following are examples of cleared events:

```
RP/0/RSP0/CPU0:Router#LC/0/1/CPU0:May 13 10:27:25.809 : g_spa_1[181]:
%L2-T1E1_LNM-3-MAJWARNNOISE :
Interface T10/1/1/0/1/1/1, noise cleared major warning threshold

RP/0/RSP0/CPU0:Router#LC/0/1/CPU0:May 13 10:28:14.810 : g_spa_1[181]:
%L2-T1E1_LNM-3-MINWARNNOISE :
Interface T10/1/1/0/1/1/1, noise cleared minor warning threshold
```

LNM Logging

When you enable syslog messages for LNM events using the **lnm syslog** command, LNM messages will appear in both the system log and in the log events buffer. You can display LNM events in the log events buffer using the **show logging events buffer bistate-alarms-set** command, and also using the **show logging** command, which are described in the *Cisco ASR 9000 Series Aggregation Services Router System Monitoring Command Reference.*

LNM supports hierarchical level alarm reporting as defined in the Telcordia (Bellcore) GR-253 standard. Hierarchical alarm reporting means that whenever a higher alarm is asserted, the lower alarm state is suppressed. When the high alarm is cleared, the lower alarm will re-assert if the condition still exists.

For LNM, this means that if a major warning threshold is continuously met or exceeded resulting in a crossed event and alarm state, then a minor warning alarm state is suppressed and returned to stable state. The minor crossed event also is removed from the bistate log. When the major warning is cleared, the minor warning alarm is asserted if the condition still exists.

Only a single crossed event for major warnings will appear in the bistate log for the controller. Therefore, you will see only a single log message for a controller if noise exists above configured threshold values.

How to Configure Clear Channel T3/E3 Controllers and Channelized T1/E1 Controllers

The T3/E3 controllers are configured in the physical layer control element of the Cisco IOS XR software configuration space. This configuration is described in the following tasks:

- Configuring a Clear Channel E3 Controller, page 425
- Modifying the Default E3 Controller Configuration, page 427
- Configuring a Clear Channel T3 Controller, page 430
- Configuring a Channelized T3 Controller, page 431
- Modifying the Default T3 Controller Configuration, page 434
- Configuring a T1 Controller, page 436
- Configuring an E1 Controller, page 440
- Configuring BERT, page 444
- Configuring Link Noise Monitoring on a T1 or E1 Channel, page 451

Configuring a Clear Channel E3 Controller

When an E3 controller is in clear channel mode, it carries a single serial interface.

The E3 controllers are configured using the E3 configuration mode.

Restrictions

- If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.
- A single SPA cannot support a mixture of T3 and E3 interfaces.

- E3 is not supported on the 4-Port Channelized T3/DS0 SPA.

SUMMARY STEPS

1. **configure**
2. **controller e3** *interface-path-id*
3. **mode serial**
4. **no shutdown**
5. **end**
   or
   commit
6. **show controllers e3** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller e3`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Specifies the E3 controller name in the notation *rack/slot/module/port* and enters E3 configuration mode.

Step 3	**`mode serial`**
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# mode serial`
		Configures the mode of the port to be clear channel serial.
		Note This step is required for the 2-Port and 4-Port Channelized T3 SPA only. The 2-Port and 4-Port Clear Channel T3/E3 SPA run in serial mode by default.

Step 4	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# no shutdown`
		Removes the shutdown configuration.
		• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

	Command or Action	Purpose

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-e3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**show controllers e3** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers e3 0/1/0/0`	(Optional) Displays information about the E3 controllers.

What to Do Next

- Modify the default configuration that is running on the E3 controller you just configured, as described in the "Modifying the Default E3 Controller Configuration" section later in this module.
- Configure a bit error rate test (BERT) on the controller to test its integrity, as described in the "Configuring BERT" section later in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module later in this document.

Modifying the Default E3 Controller Configuration

This task explains how to modify the default E3 controller configuration, which is described in the "Default Configuration Values for T3 and E3 Controllers" section earlier in this module.

Prerequisites

You must configure a clear channel E3 controller, as described in the "Configuring a Clear Channel E3 Controller" section earlier in this module.

Restrictions

- E3 is not supported on the 4-Port Channelized T3/DS0 SPA.

SUMMARY STEPS

1. **configure**
2. **controller e3** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **cablelength** *feet*
5. **framing** {**g751** | **g832**}
6. **national bits** {**disable** | **enable**}
7. **no shutdown**
8. **end**
or
commit
9. **show controllers e3** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller e3`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Specifies the E3 controller name in the notation *rack/slot/module/port* and enters E3 configuration mode.

Step 3	**`clock source`** `{`**`internal`** `|` **`line`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# clock source internal`
		(Optional) Sets the clocking for individual E3 links.
		Note The default clock source is **internal**.
		Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Step 4	**`cablelength`** *`feet`*
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# cablelength 250`
		(Optional) Specifies the distance of the cable from the router to the network equipment.
		Note The default cable length is 224 feet.

	Command or Action	Purpose

Step 5	**`framing`** `{`**`g751`** `|` **`g832`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# framing g832`
		(Optional) Selects the frame type for the E3 port. Possible E3 frame types are G.751 and G.832.
		Note The default framing for E3 is G.751.

Step 6	**`national bits`** `{`**`disable`** `|` **`enable`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# national bits enable`
		(Optional) Enables or disables the 0x1F national reserved bit pattern on the E3 port.
		Note The E3 national bit is enabled by default, and the bit pattern value is 1.

Step 7	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# no shutdown`
		Removes the shutdown configuration.
		• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

Step 8	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-e3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 9	**`show controllers e3`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show controllers e3 0/1/0/0`
		(Optional) Displays information about the E3 controllers.

What to Do Next

- Modify the default configuration that is running on the T3 controller you just configured, as described in the "Modifying the Default T3 Controller Configuration" section later in this module.
- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section later in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module later in this document.

Configuring a Clear Channel T3 Controller

When a T3 controller is in clear channel mode, it carries a single serial interface.

The T3 controllers are configured in the T3 configuration mode.

Prerequisites

Before you can configure a clear channel T3 controller on a channelized SPA, you must configure the SPA for an STS stream channelized for T3. For more information, see the "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" module.

Restrictions

- If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.
- A single SPA cannot support a mixture of T3 and E3 interfaces.

SUMMARY STEPS

1. **configure**
2. **controller t3** *interface-path-id*
3. **mode serial**
4. **no shutdown**
5. **end**
   or
   commit
6. **show controllers t3** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller t3** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`	Specifies the T3 controller name in the *rack/slot/module/port* notation and enters T3 configuration mode.
Step 3	**mode serial** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# mode serial`	**Note** Configures the mode of the port to be clear channel serial.

	Command or Action	Purpose

Step 4	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# no shutdown`
		Removes the shutdown configuration.
		• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

Step 5	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**`show controllers t3`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show controllers t3 0/1/0/0`
		(Optional) Displays information about the T3 controllers.

What to Do Next

- Modify the default configuration that is running on the T3 controller you just configured, as described in the "Modifying the Default T3 Controller Configuration" section later in this module.
- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section later in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module.

Configuring a Channelized T3 Controller

The SPAs that support channelized T3 support channelization to T1, E1, and DS0. The steps in this section describe how to channelize a single T3 controller into 28 T1 controllers or 21 E1 controllers. Once you have created T1 or E1 controllers, you can further channelize those controllers into DS0 time slots, as described in the following sections:

- Configuring a T1 Controller

- Configuring an E1 Controller

Each individual T1 controller supports a total of 24 DS0 time slots, and each individual E1 controller supports a total of 31 DS0 time slots.

Prerequisites

Before you configure a channelized T3 controller, be sure that the following requirements are met:

- You have one of the following SPAs installed:
  - 1-Port Channelized OC-3/STM-1 SPA
  - 2-Port Channelized OC-12/DS0 SPA
  - 4-Port Channelized T3/DS0 SPA
- For the channelized SONET SPAs, you have configured the SPA for an STS stream channelized for T3. For more information, see the "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" module.

Note If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.

SUMMARY STEPS

1. **configure**
2. **controller t3** *interface-path-id*
3. **mode** [**t1** | **e1**]
4. **no shutdown**
5. **end**
or
commit
6. **show controllers t3** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller T3`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Specifies the T3 controller name in the notation *rack/slot/module/port* and enters T3 configuration mode.

	Command or Action	Purpose
Step 3	**mode t1** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# mode t1`	Sets the mode of the channelized controllers to be T1, and creates 28 T1 controllers.
Step 4	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# no shutdown`	Removes the shutdown configuration. • The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**show controllers t3** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers t3 0/1/0/0`	(Optional) Displays information about the T3 controllers.

What to Do Next

- Modify the default configuration that is running on the T3 controller you just configured, as described in the "Modifying the Default T3 Controller Configuration" section on page 434.
- If you channelized your T3 controller into 28 T1 controllers, configure the T1 controllers and assign DS0 time slots to them, as described in the *"Configuring a T1 Controller" section on page 436*.
- If you channelized your T3 controller into 21 E1 controllers, configure the E1 controllers and assign DS0 time slots to them, as described in the *"Configuring an E1 Controller" section on page 440*.

Modifying the Default T3 Controller Configuration

This task explains how to modify the default T3 controller configuration, which is described in the "Default Configuration Values for T3 and E3 Controllers" section on page 422.

Prerequisites

You must configure a clear channel or channelized T3 controller, as described in one of the following sections:

- Configuring a Clear Channel T3 Controller
- Configuring a Channelized T3 Controller

SUMMARY STEPS

1. **configure**
2. **controller t3** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **cablelength** *feet*
5. **framing** {**auto-detect** | **c-bit** | **m23**}
6. **mdl transmit** {**idle-signal** | **path** | **test-signal**} {**disable** | **enable**}
7. **mdl string** {**eic** | **fi** | **fic** | **gen-number** | **lic** | **port-number** | **unit**} *string*
8. **no shutdown**
9. **end**
   or
   commit
10. **show controllers t3** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`controller T3 interface-path-id`
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Specifies the T3 controller name in the notation *rack/slot/module/port* and enters T3 configuration mode.

	Command or Action	Purpose

Step 3	**clock source** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# clock source internal`
		(Optional) Sets the clocking for the T3 port.
		Note The default clock source is **internal**.
		Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Step 4	**cablelength** *feet* **Example:** `RP/0/RSP0/CPU0:router(config-t3)# cablelength 250`	(Optional) Specifies the distance of the cable from the router to the network equipment. **Note** The default cable length is 224 feet.

Step 5	**framing** {**auto-detect** | **c-bit** | **m23**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# framing c-bit`
		(Optional) Selects the frame type for the T3 port.
		Note The default frame type for T3 is C-bit. Auto-detect is not supported on the 2-Port Channelized OC-12c/DS0 SPA.

Step 6	**mdl transmit** {**idle-signal** | **path** | **test-signal**} {**disable** | **enable**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# mdl transmit path enable`
		(Optional) Enables Maintenance Data Link (MDL) messages on the T3 port.
		Note MDL messages are supported only when the T3 framing is C-bit parity.
		Note MDL message are disabled by default.

Step 7	**mdl string** {**eic** | **fi** | **fic** | **gen-number** | **lic** | **port-number** | **unit**} *string* **Example:** `RP/0/RSP0/CPU0:router(config-t3)# mdl fi facility identification code`	(Optional) Specifies the values of the strings sent in the MDL messages.
Step 8	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# no shutdown`	Removes the shutdown configuration. • The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

	Command or Action	Purpose

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 10	**show controllers t3** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers t3 0/1/0/0`	(Optional) Displays information about the T3 controllers.

What to Do Next

- If you configured a clear channel T3 controller, perform the following tasks:
  - Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section on page 444 later in this module.
  - Configure the associated serial interface, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module.
- If you channelized your T3 controller into 28 T1 controllers, configure the T1 controllers and assign DS0 time slots to them, as described in the *"Configuring a T1 Controller" section on page 436*.
- If you channelized your T3 controller into 21 E1 controllers, configure the E1 controllers and assign DS0 time slots to them, as described in the *"Configuring an E1 Controller" section on page 440*.

Configuring a T1 Controller

This task describes how to configure an individual T1 controller and channelize it into 24 individual DS0 timeslots.

Prerequisites

Before you configure a T1 controller, be sure that the following requirements are met:

- You have one of the following SPAs installed:
  - 1-Port Channelized OC-3/STM-1 SPA
  - 2-Port Channelized OC-12/DS0 SPA
  - 4-Port Channelized T3/DS0 SPA
  - 8-Port Channelized T1/E1 SPA
- If you have a 1-Port Channelized OC-3/STM-1 SPA or 2-Port Channelized OC-12/DS0 SPA, you must complete the following configuration:
  - Configure an STS stream channelized for T3. For more information, see the "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" module.
  - Configure a channelized T3 controller running in T1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 431.
- If you have a 4-Port Channelized T3/DS0 SPA, you must configure a channelized T3 controller to run in T1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 431.

Restrictions

If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.

Before you configure a T1 controller on the 8-Port Channelized T1/E1 SPA, consider the following restrictions:

- The SPA controller is not visible until it is explicitly configured for T1 mode.
- For each individual SPA, the SPA ports must all be in the same mode (all T1).

SUMMARY STEPS

1. **show controllers t1** *interface-path-id*
2. **configure**
3. **controller t1** *interface-path-id*
4. **framing** {**sf** | **esf**}
5. **yellow** {**detection** | **generation**} {**disable** | **enable**}
6. **clock source** {**internal** | **line**}
7. **fdl** {**ansi** | **att**} {**enable** | **disable**}
8. **no shutdown**
9. **channel-group** *channel-group-number*
10. **timeslots** *range*
11. **speed** *kbps*
12. **exit**
13. Repeat Step 9 through Step 12 to assign time slots to a channel group. Each controller can contain up to 24 time slots.
14. **exit**
15. Repeat Step 2 through Step 14 to assign more channel groups to a controller.

16. **end**
    or
    commit

DETAILED STEPS

Step 1	**show controllers t1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers t3 0/1/0/0`	(Optional) Displays information about the T1 controllers you created in Step 3.
Step 2	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**controller t1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/3/0/0/0`	Enters T1 configuration mode.

Step 4	**framing** {**sf** | **esf**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# framing esf`
		(Optional) Selects the frame type for the T1 data line:
		• **sf**—Superframe
		• **esf**—Extended super frame
		Note The default frame type for T1 is Extended superframe (**esf**).

Step 5	**yellow** {**detection** | **generation**} {**disable** | **enable**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t1e1)# yellow detection enable`
		(Optional) Enables or disables the detection and generation of T1 yellow alarms.
		Note Yellow alarms are detected and generated on the T1 channel by default.

Step 6	**clock source** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t1e1)# clock source internal`
		(Optional) Sets the clocking for individual T1 links.
		Note The default clock source is **internal**.
		Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Step 7	**fdl** {**ansi** | **att**} {**enable** | **disable**}
	`Example:`
	`RP/0/RSP0/CPU0:router(config-t1e1)# fdl ansi enable`
		(Optional) Enables the transmission of ANSI T1.403 or AT&T TR54016 once-per-second performance reports through Facility Data Link (FDL).
		Note FDL ansi and att are disabled by default.

Step 8	**`no shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t1e1)# no shutdown`
		Removes the shutdown configuration.
		• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

Step 9	**`channel-group`** *`channel-group-number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# channel-group 0`
		Creates a T1 channel group and enters channel group configuration mode for that channel group.

Step 10	**`timeslots`** *`range`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 7-12`
		Associates DS0 time slots to a channel group and creates an associated serial subinterface on that channel group.
		• Range is from 1 to 24 time slots.
		• You can assign all 24 time slots to a single channel group, or you can divide the time slots among several channel groups.
		Note Each individual T1 controller supports a total of 24 DS0 time slots.

Step 11	**`speed`** *`kbps`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1e1-channel_group)# speed 64`
		(Optional) Specifies the speed of the DS0s in kilobits per second. Valid values are 56 and 64.
		Note The default speed is 64 kbps.

Step 12	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit`
		Exits channel group configuration mode.

Step 13	Repeat Step 9 through Step 12 to assign time slots to a channel group. Each controller can contain up to 24 time slots.	—

Step 14	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# exit`
		Exits T1 configuration mode and enters global configuration mode.

Step 15	Repeat Step 2 through Step 14 to assign more channel groups to a controller as desired.	—

Step 16	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

What to Do Next

- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section on page 444.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module.

Configuring an E1 Controller

This task describes how to configure an individual E1 controller and channelize it into 31 individual DS0 timeslots.

Prerequisites

Before you configure an E1 controller, be sure that the following requirements are met:

- You have one of the following SPAs installed:
  - 1-Port Channelized OC-3/STM-1 SPA
  - 2-Port Channelized OC-12/DS0 SPA
  - 4-Port Channelized T3/DS0 SPA
  - 8-Port Channelized T1/E1 SPA
- If you have a 1-Port Channelized OC-3/STM-1 SPA or 2-Port Channelized OC-12/DS0 SPA, you must complete the following configuration:

  - Configure an STS stream channelized for T3. For more information, see the "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" module.
  - Configure a channelized T3 controller running in E1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 431.
- If you have a 4-Port Channelized T3/DS0 SPA, you must configure a channelized T3 controller to run in E1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 431.

Restrictions

If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.

Before you configure an E1 controller on the 8-Port Channelized T1/E1 SPA, consider the following restrictions:

- The SPA controller is not visible until it is explicitly configured for E1 mode.
- For each individual SPA, the SPA ports must all be in the same mode (all E1).

SUMMARY STEPS

1. **show controllers e1** *interface-path-id*
2. **configure**
3. **controller e1** *interface-path-id*
4. **clock source** {**internal** | **line**}
5. **framing** {**crc4** | **no-crc4** | **unframed**}
6. **national bits** *bits*
7. **no shutdown**
8. **channel-group** *channel-group-number*
9. **timeslots** *range*
10. **speed** *kbps*
11. **exit**
12. Repeat Step 8 through Step 11 to assign time slots to a channel group. Each controller can contain up to 24 time slots.
13. **exit**
14. Repeat Step 2 through Step 13 to assign more channel groups to a controller as desired.
15. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show controllers e1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers e1 0/1/0/0`	(Optional) Displays information about the E1 controllers.
Step 2	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**controller e1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller e1 0/3/0/0/0`	Enters E1 configuration mode.

Step 4	**clock source** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-e1)# clock source internal`
		(Optional) Sets the clocking for individual E1 links.
		Note The default clock source is **internal**.
		Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Step 5	**framing** {**crc4** | **no-crc4** | **unframed**}
	Example:
	`RP/0/RSP0/CPU0:router(config-e1)# framing unframed`
		(Optional) Selects the frame type for the E1 data line. The following frame types are valid for E1:
		• **crc4**—Framing with CRC-4 error monitoring capabilities
		• **no-crc4**—Framing without CRC-4 error monitoring capabilities
		• **unframed**—Unframed E1
		Note The default frame type for E1 is **crc4**.

Step 6	**national bits** *bits* **Example:** `RP/0/RSP0/CPU0:router(config-e1)# national bits 10`	(Optional) Specifies the national reserved bits for an E1 port. Range is from 0 to 31. **Note** The default bit pattern is 0, which corresponds to the hexadecimal value *0x1f*.
Step 7	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router(config-e1)# no shutdown`	Removes the shutdown configuration. • The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

	Command or Action	Purpose
Step 8	**channel-group** *channel-group-number* **Example:** `RP/0/RSP0/CPU0:router(config-e1)# channel-group 0`	Creates an E1 channel group and enters channel group configuration mode for that channel group.
Step 9	**timeslots** *range* **Example:** `RP/0/RSP0/CPU0:router(config-e1-channel_group)# timeslots 1-16`	Associates one or more time slots to a channel group and creates an associated serial subinterface on that channel group. • Range is from 1 to 31 time slots. • You can assign all 31 time slots to a single channel group, or you can divide the time slots among several channel groups. **Note** Each E1 controller supports a total of 31 DS0 time slots.
Step 10	**speed** *kbps* **Example:** `RP/0/RSP0/CPU0:router(config-e1-channel_group)# speed 100`	(Optional) Specifies the speed of the DS0s in kilobits per second. Valid values are 56 and 64. **Note** The default speed is 64 kbps.
Step 11	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-e1-channel_group)# exit`	Exits channel group configuration mode
Step 12	Repeat Step 8 through Step 11 to assign time slots to a channel group.	—
Step 13	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-e1)# exit`	Exits E1 configuration mode

	Command or Action	Purpose
Step 14	Repeat Step 2 through Step 13 to assign more channel groups to a controller as desired.	—

Step 15	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-e3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-e3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

What to Do Next

- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section on page 444 in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module later in this document.

Configuring BERT

Depending on your hardware support, BERT is supported on each of the T3/E3 or T1/E1 controllers, and on the DS0 channel groups. It is done only over an unframed T3/E3 or T1/E1 signal and is run on only one port at a time. It is also supported on individual channel groups.

To view the BERT results, use the **show controllers t1** or **show controllers t3** command in EXEC mode. The BERT results include the following information:

- Type of test pattern selected
- Status of the test
- Interval selected
- Time remaining on the BER test
- Total bit errors
- Total bits received

BERT is data intrusive. Regular data cannot flow on a line while the test is in progress. The line is put in an alarm state when BERT is in progress and restored to a normal state after BERT has been terminated.

Configuring BERT on T3/E3 and T1/E1 Controllers

This task explains how to enable a bit error rate test (BERT) pattern on a T3/E3 or T1/E1 line or an individual channel group.

Prerequisites

You must have configured a clear channel T3/E3 controller, or a channelized T3-to-T1/E1 controller.

Restrictions

Before configuring BERT on the 1-Port Channelized OC-48/STM-16 SPA, consider the following restrictions:

- Only two simultaneous BERT tests are possible per STS-12 stream.
- These test patterns are supported:
  - 2^15-1 (O.151)
  - 2^20-1 (O.151) - QRSS
  - 2^23-1 (O.151)
  - Fixed Patterns (all 0s, all 1s etc.)
  - Single bit error injection
  - Data inversion

Before configuring BERT on the 4-Port Channelized T3/DS0 SPA, consider the following restrictions:

- A maximum of 12 BERT sessions is supported.
- 6 simultaneous BERT sessions among the first three physical ports and 6 simultaneous BERT sessions on the fourth port are supported.
- Only one BERT session per T1 is supported.
- These test patterns are supported on the 4-Port Channelized T3/DS0 SPA:
  - 2^11-1—T1/E1/DS0 only
  - 2^15-1 (O.151)
  - 2^20-1 (O.153)—T3 only
  - 2^20-1 (QRSS)
  - 2^23-1 (O.151)
  - Alternating 0s/1s
  - Fixed Patterns (all 0s, all 1s etc.)
  - 1 in 8 DS1 insertion—T1/E1/DS0 only
  - 3 in 24 DS1 insertion—T1/E1/DS0 only

These test patterns are supported on the 8-Port Channelized T1/E1 SPA for T1/E1/DS0:

- 2^11-1
- 2^15-1 (O.153)
- 2^20-1 (QRSS)
- 2^23-1 (O.151)
- Alternating 0s/1s
- Fixed Patterns (all 0s, all 1s etc.)

For other cards, valid patterns for all controllers and channel groups include: 0s, 1s, 2^15, 2^20, 2^20-QRSS, 2^23, and alt-0-1.

Additional valid patterns for T1 and E1 controllers include: 1in8, 3in24, 55Daly, and 55Octet. Additional valid patterns for channel groups include: 2^11, 2^9, ds0-1, ds0-2, ds0-3, and ds0-4.

SUMMARY STEPS

1. **configure**
2. **controller** [**t3** | **e3** | **t1** | **e1**] *interface-path-id*
3. **pattern** *pattern*
4. **bert interval** *time*
5. **bert error** [*number*]
6. **end**
   or
   commit
7. **exit**
8. **exit**
9. **bert** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] [**error**] **start**
10. **bert** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop**
11. **show controllers** [**t3** | **e3** | **t1** | **e1**] *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`controller [t3 | e3 | t1 | e1] interface-path-id`
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Specifies the controller name and instance in the notation *rack/slot/module/por*t, and enters T3, E3, T1, or E1 controller configuration mode.

	Command or Action	Purpose

Step 3	**`bert pattern`** *`pattern`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# bert pattern 2^15`
		Enables a specific bit error rate test (BERT) pattern on a controller.
		Note You must use the **bert** command in EXEC mode to start the BER test.

Step 4	**`bert interval`** *`time`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# bert pattern 2^15`
		(Optional) Specifies the duration of a bit error rate test (BERT) pattern on a T3/E3 or T1/E1 line. The interval can be a value from 1 to 14400.

Step 5	**`bert error`** `[`*`number`*`]`
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# bert error 10`
		Specifies the number of BERT errors to introduce into the bit stream. Range is from 1 to 255.

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 7	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# exit`
		Exits T3/E3 or T1/E1 controller configuration mode.

Step 8	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# exit`
		Exits global configuration mode.

	Command or Action	Purpose

Step 9	**bert** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] [**error**] **start**
	Example:
	`RP/0/RSP0/CPU0:router# bert t3 0/3/0/0 start`
	`RP/0/RSP0/CPU0:router# bert t3 0/3/0/0 error`
		Starts the configured BERT test on the specified T3/E3 or T1/E1 controller.
		Note You can include the optional **error** keyword to inject errors into the running BERT stream.

Step 10	**bert** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop** **Example:** `RP/0/RSP0/CPU0:router# bert t3 0/3/0/0 stop`	Stops the configured BERT test on the specified T3/E3 or T1/E1 controller.
Step 11	**show controllers** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers t3 0/3/0/0`	Displays the results of the configured BERT.

What to Do Next

Configure the serial interfaces that are associate with the controllers you tested, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module.

Configuring BERT on a DS0 Channel Group

This task explains how to enable a bit error rate test (BERT) pattern on an individual DS0 channel group.

Prerequisites

You must have configured a clear channel T1/E1 controller, or a channelized T3-to-T1/E1 controller.

SUMMARY STEPS

1. **configure**
2. **controller** {**t1** | **e1**} *interface-path-id*
3. **channel-group** *channel-group-number*
4. **bert pattern** *pattern*
5. **bert interval** *time*
6. **end**
   or
   commit
7. **exit**
8. **exit**
9. **exit**
10. **bert** [**t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*][**error**] **start**

11. **bert** [**t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop**
12. **show controllers** [**t1** | **e1**] *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller** {**t1** | **e1**} *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`	Specifies the controller name and instance in the notation *rack/slot/module/por*t, and enters T1 or E1 controller configuration mode.

Step 3	**channel-group** *channel-group-number*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# channel-group 1`
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)#`
		Enters channel group configuration mode for a specific channel group. Replace *channel-group-number* with the number that identifies the channel group on which you want to configure a BERT.

Step 4	**bert pattern** *pattern* **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# bert pattern 2^15`	Enables a specific bit error rate test (BERT) pattern on a T1 line. Valid patterns for all controllers and channel groups include: **0s**, **1s**, **2^15**, **2^20**, **2^20-QRSS**, **2^23**, and **alt-0-1**. Additional valid patterns for T1 and E1 controllers include: **1in8**, **3in24**, **55Daly**, and **55Octet**. Additional valid patterns for channel groups include: **2^11**, **2^9**, **ds0-1**, **ds0-2**, **ds0-3**, and **ds0-4.** **Note** You must use the **bert** command in EXEC mode to start the BER test.
Step 5	**bert interval** *time* **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# bert interval 5`	(Optional) Specifies the duration, in minutes, of a bit error rate test (BERT) pattern on a T1/E1 line. The interval can be a value from 1 to 14400.

	Command or Action	Purpose

Step 6	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 7	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit`	Exits channel group configuration mode.
Step 8	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t1)# exit`	Exits T1 or E1 configuration mode.
Step 9	**exit** **Example:** `RP/0/RSP0/CPU0:router(config)# exit`	Exits global configuration mode.

Step 10	**bert** [**t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] [**error**] **start**
	Example:
	`RP/0/RSP0/CPU0:router# bert t1 0/3/0/0/0 start`
	`RP/0/RSP0/CPU0:router# bert t1 0/3/0/0/0 error`
		Starts the configured BERT test on the specified channel group.
		Note You can include the optional **error** keyword to inject errors into the running BERT stream.

	Command or Action	Purpose
Step 11	**bert** [**t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop** **Example:** `RP/0/RSP0/CPU0:router# bert t1 0/3/0/0/0 stop`	Stops the configured BERT test on the specified channel group.
Step 12	**show controllers** [**t1** | **e1**] *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show controllers t3 0/3/0/0`	Displays the results of the configured BERT.

What to Do Next

Configure the serial interfaces that are associate with the controllers you tested, as described in the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module later in this document.

Configuring Link Noise Monitoring on a T1 or E1 Channel

This section describes how to configure Link Noise Monitoring (LNM) on a T1 or E1 channel on the Cisco ASR 9000 Series Router.

Prerequisites

Before you configure LNM on the Cisco ASR 9000 Series Router, be sure that these requirements are met:

- A 2-Port Channelized OC-12c/DS0 SPA is installed.
- The 2-Port Channelized OC-12/DS0 SPA is configured as a channelized T3 controller running in T1 or E1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 431.
- The T1 or E1 controller is configured as a single channel supporting the full 24 or 31 DS0 time slots, as described in the "Configuring a T1 Controller" section on page 436 or "Configuring an E1 Controller" section on page 440. LNM is not supported on a fractional T1 or E1 link.

Restrictions

Before you configure LNM on the Cisco ASR 9000 Series Router, consider these restrictions:

- The **lnm major-warning** and **lnm remove** commands are mutually exclusive. You can only configure one of these LNM functions on a controller.
- The **lnm minor-warning** command can be configured with the **lnm major-warning** or **lnm remove** commands on a controller.
- When the **lnm remove** command is configured, links in an MLPPP bundle are removed only up to the threshold set by the **ppp multilink minimum-active links** command.

SUMMARY STEPS

1. **configure**

2. **controller** {**t1** | **e1**} *interface-path-id*
3. **lnm {major-warning** | **remove} [clear** | **set**][**line-code-violation** *lcv-value* [**path-code-violation** *pcv-value*]][**duration** *seconds*]
4. **lnm minor-warning** [**clear** | **set**][**line-code-violation** *lcv-value* [**path-code-violation** *pcv-value*]][**duration** *seconds*]
5. **lnm syslog**
6. **end**
   or
   commit

DETAILED STEPS

Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller** {**t1** | **e1**} *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/1/1/0/1/1`	Enters T1 or E1 configuration mode.
Step 3	**lnm** {**major-warning** | **remove**}[**clear** | **set**][**line-code-violation** *lcv-value* [**path-code-violation** *pcv-value*]][**duration** *seconds*] **Example:** `RP/0/RSP0/CPU0:router(config-t1)# lnm major-warning`	(Optional) Enables link noise monitoring and specifies thresholds for noise errors on T1/E1 links that are used to signal major warning events or link removal and recovery from those events. The default values for both set and clear thresholds are: • For T1 links—**line-code-violation** is 1544, **path-code-violation** is 320, and **duration** is 10. • For E1 links—**line-code-violation** is 2048, **path-code-violation** is 831, and **duration** is 10.
Step 4	**lnm minor-warning** [**clear** | **set**][**line-code-violation** *lcv-value* [**path-code-violation** *pcv-value*]][**duration** *seconds*] **Example:** `RP/0/RSP0/CPU0:router(config-t1)# lnm minor-warning`	(Optional) Enables link noise monitoring and specifies thresholds for noise errors on T1/E1 links that are used to signal minor warning events and recovery from those events. The default values for both set and clear thresholds are: • For T1 links—**line-code-violation** is 154, **path-code-violation** is 145, and **duration** is 10. • For E1 links—**line-code-violation** is 205, **path-code-violation** is 205, and **duration** is 10.

Step 5 **lnm syslog**

Example:

```
RP/0/RSP0/CPU0:router(config-t1)# lnm syslog
```

(Optional) Enables logging of link noise monitoring major and minor events and alarms.

Note You must use this command for LNM messages to appear in both the system log and in the log events buffer.

Step 6 **end**

or

commit

Example:

```
RP/0/RSP0/CPU0:router(config-t1)# end
```

or

```
RP/0/RSP0/CPU0:router(config-t1)# commit
```

Saves configuration changes.

- When you issue the **end** command, the system prompts you to commit changes:

  ```
  Uncommitted changes found, commit them before
  exiting(yes/no/cancel)?
  [cancel]:
  ```

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Verifying Link Noise Monitoring Configuration and Status

To verify LNM configuration and state information, as well as statistics and events, use the **show controllers lnm** command as shown in the following example:

Note When the **lnm remove** command is configured, the word "Remove" appears in the show controllers output headers and events in place of "major-warning" and "Major-Warn."

```
RP/0/RSP0/CPU0:Router# show controllers t1 0/1/1/0/1/1 lnm all
Thu May 13 10:28:26.474 PDT

Controller T1 0/1/1/0/1/1

Syslog   Monitoring type  State     Thresholds (lcv/pcv/duration)
--------------------------------------------------------------------
enabled   minor-warning   stable    Set(  15/  15/   4) Clear(  15/  15/   4)
          major-warning   stable    Set( 154/ 145/   4) Clear( 154/ 145/   4)

  Monitoring type           Minor-Warn      Major-Warn
  ---------------           ------------    ------------
  Create                    1               1
  Update                    0               0
  Delete                    0               0
  Clear                     0               0
  Noise Crossed             1               1
  Noise Cleared             1               1
```

```
Last Five Events
--------------------------------------------------------------------
MINWARNCROSS: Noise crossed minor-warn threshold at Thu May 13 09:54:10 2010
MAJWARNCROSS: Noise crossed major-warn threshold at Thu May 13 09:54:11 2010
MAJWARNCLEAR: Noise cleared major-warn threshold at Thu May 13 10:27:25 2010
MINWARNCLEAR: Noise cleared minor-warn threshold at Thu May 13 10:28:14 2010
```

Clearing Link Noise Monitoring States and Statistics

You can use the **clear controller lnm** command to reset LNM states or clear statistics and reset them to zero.

There should not normally be any need to clear the LNM controller states. The **state** option resets the LNM configuration which causes an update of the current LNM states in the system. Therefore, under normal conditions, if the controller is in alarm state, the reset should continue to report the alarm state; alternatively, if the controller is clear of any alarms, the reset will show the stable state. The use of the **clear controller lnm state** command does not actually clear any alarms, but causes a refresh of their values in the system. Therefore, this command can be used if the reported controller state should happen to be out of synchronization with the actual controller state.

To reset LNM states, use the **clear controller lnm** command as shown in the following example:

```
RP/0/RSP0/CPU0:Router# clear controller t1 0/1/0/0/1/1 lnm state
```

To clear LNM statistics and reset counters to zero, use the **clear controller lnm** command as shown in the following example:

```
RP/0/RSP0/CPU0:Router# clear controller t1 0/1/0/0/1/1 lnm statistics

RP/0/RSP0/CPU0:Router# show controller T1 0/1/0/1/1/1 lnm statistics
Thu May 13 11:26:20.991 PDT

Controller T1 0/1/0/1/1/1

  Monitoring type            Minor-Warn      Major-Warn
  ---------------            ------------    ------------
  Create                     0               0
  Update                     0               0
  Delete                     0               0
  Clear                      0               0
  Noise Crossed              0               0
  Noise Cleared              0               0
```

Configuration Examples

This section contains the following examples:

- Configuring a Clear Channel T3 Controller: Example, page 455
- Configuring a T3 Controller with Channelized T1 Controllers: Example, page 455
- Configuring BERT on a T3 Controller: Example, page 456
- Configuring Link Noise Monitoring on a T1 Controller: Examples, page 457
- QoS on T3 Channels: Example, page 458

Configuring a Clear Channel T3 Controller: Example

The following example shows configuration for a clear channel T3 controller:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)#controller T3 0/3/2/0
RP/0/RSP0/CPU0:router(config-t3)#clock source internal
RP/0/RSP0/CPU0:router(config-t3)#mode serial
RP/0/RSP0/CPU0:router(config-t3)#cablelength 4
RP/0/RSP0/CPU0:router(config-t3)#framing c-bit
RP/0/RSP0/CPU0:router(config-t3)#commit
```

Configuring a T3 Controller with Channelized T1 Controllers: Example

The following example shows how to configure a T3 controller that has been channelized 28 T1 controllers:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# controller T3 0/3/0/0
RP/0/RSP0/CPU0:router(config-t3)# mode t1
RP/0/RSP0/CPU0:router(config-t3)# framing m23
RP/0/RSP0/CPU0:router(config-t3)# cablelength 11
RP/0/RSP0/CPU0:router(config-t3)# clock source line
RP/0/RSP0/CPU0:router(config-t3)#commit
RP/0/RSP0/CPU0:router(config-t3)#exit
RP/0/RSP0/CPU0:router(config)# exit
RP/0/RSP0/CPU0:router# show controllers T1 ?

  0/3/0/0/0   T1 Interface Instance
  0/3/0/0/1   T1 Interface Instance
  0/3/0/0/10  T1 Interface Instance
  0/3/0/0/11  T1 Interface Instance
  0/3/0/0/12  T1 Interface Instance
  0/3/0/0/13  T1 Interface Instance
  0/3/0/0/14  T1 Interface Instance
  0/3/0/0/15  T1 Interface Instance
  0/3/0/0/16  T1 Interface Instance
  0/3/0/0/17  T1 Interface Instance
  0/3/0/0/18  T1 Interface Instance
  0/3/0/0/19  T1 Interface Instance
  0/3/0/0/2   T1 Interface Instance
  0/3/0/0/20  T1 Interface Instance
  0/3/0/0/21  T1 Interface Instance
  0/3/0/0/22  T1 Interface Instance
  0/3/0/0/23  T1 Interface Instance
  0/3/0/0/24  T1 Interface Instance
  0/3/0/0/25  T1 Interface Instance
  0/3/0/0/26  T1 Interface Instance
  0/3/0/0/27  T1 Interface Instance
  0/3/0/0/3   T1 Interface Instance
  0/3/0/0/4   T1 Interface Instance
  0/3/0/0/5   T1 Interface Instance
 --More--
!
RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router(config)#configure
RP/0/RSP0/CPU0:router(config)# controller t1 0/3/0/0/0
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# exit
```

```
RP/0/RSP0/CPU0:router(config)# controller t1 0/3/0/0/1
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# exit
RP/0/RSP0/CPU0:router(config)# controller t1 0/3/0/0/2
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-12
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# channel-group 1
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 13-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# exit
RP/0/RSP0/CPU0:router(config)# controller t1 0/3/0/0/3
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-6
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# channel-group 1
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 7-12
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# channel-group 2
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 13-18
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# channel-group 3
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 19-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1-channel_group)#commit
```

Configuring BERT on a T3 Controller: Example

The following example shows how to configure a BERT on a T3 controller, and then display the results of the BERT:

```
RP/0/RSP0/CPU0:router# config
RP/0/RSP0/CPU0:router(config)# controller t3 0/3/0/1
RP/0/RSP0/CPU0:router(config-t3)# bert pattern 0s

Run bert from exec mode for the bert config to take effect

RP/0/RSP0/CPU0:router(config-t3)#exit
RP/0/RSP0/CPU0:router(config)# exit

Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]
RP/0/RSP0/CPU0:router# bert t3 0/3/0/1 start


RP/0/RSP0/CPU0:router# bert t3 0/3/0/1 stop


RP/0/RSP0/CPU0:router# show controllers t3 0/3/0/1

  T30/3/0/1 is up
  No alarms detected.
  MDL transmission is disabled
    EIC: , LIC: , FIC: , UNIT:
     Path FI:
     Idle Signal PORT_NO:
     Test Signal GEN_NO:
  FEAC code received: No code is being received
  Framing is C-BIT Parity, Line Code is B3ZS, Clock Source is Internal
  Data in current interval (108 seconds elapsed):
     0 Line Code Violations, 0 P-bit Coding Violation
```

```
      0 C-bit Coding Violation, 0 P-bit Err Secs
      0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
      0 Unavailable Secs, 0 Line Errored Secs
      0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
   Data in Interval 1:
      0 Line Code Violations, 0 P-bit Coding Violation
      0 C-bit Coding Violation, 0 P-bit Err Secs
      0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
      0 Unavailable Secs, 0 Line Errored Secs
      0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
   Data in Interval 2:
      0 Line Code Violations, 0 P-bit Coding Violation
      0 C-bit Coding Violation, 0 P-bit Err Secs
      0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
      0 Unavailable Secs, 0 Line Errored Secs
      0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
   Data in Interval 3:
      0 Line Code Violations, 0 P-bit Coding Violation
      0 C-bit Coding Violation, 0 P-bit Err Secs
      0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
      0 Unavailable Secs, 0 Line Errored Secs
      0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
```

Configuring Link Noise Monitoring on a T1 Controller: Examples

The following example shows how to configure a channelized T3 controller for T1 configuration mode using the full 24 DS0 timeslots as a single channel before configuring LNM on the link. In this example, the values shown are actually the system defaults for the set thresholds:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# controller T3 0/1/1/0/1
RP/0/RSP0/CPU0:router(config-t3)# mode t1
RP/0/RSP0/CPU0:router(config-t3)# framing m23
RP/0/RSP0/CPU0:router(config-t3)# cablelength 11
RP/0/RSP0/CPU0:router(config-t3)# clock source line
RP/0/RSP0/CPU0:router(config-t3)#commit
RP/0/RSP0/CPU0:router(config-t3)#exit
RP/0/RSP0/CPU0:router(config)# controller t1 0/1/1/0/1/1
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# lnm syslog
RP/0/RSP0/CPU0:router(config-t1)# lnm major-warning set line-code-violation 1544
path-code-violation 320 duration 10
RP/0/RSP0/CPU0:router(config-t1)# lnm minor-warning set line-code-violation 154
path-code-violation 145 duration 10
```

The following example shows how to configure a channelized T3 controller for T1 configuration mode using the full 24 DS0 timeslots as a single channel before configuring LNM on the link. In this example, the values shown are actually the system defaults for the set thresholds, and LNM is configured to signal the Noise Attribute to PPP for MLPPP link removal when those thresholds are crossed:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# controller T3 0/1/1/0/1
RP/0/RSP0/CPU0:router(config-t3)# mode t1
RP/0/RSP0/CPU0:router(config-t3)# framing m23
RP/0/RSP0/CPU0:router(config-t3)# cablelength 11
RP/0/RSP0/CPU0:router(config-t3)# clock source line
RP/0/RSP0/CPU0:router(config-t3)#commit
RP/0/RSP0/CPU0:router(config-t3)#exit
RP/0/RSP0/CPU0:router(config)# controller t1 0/1/1/0/1/1
```

```
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# lnm syslog
RP/0/RSP0/CPU0:router(config-t1)# lnm remove set line-code-violation 1544
path-code-violation 320 duration 10
RP/0/RSP0/CPU0:router(config-t1)# lnm minor-warning set line-code-violation 154
path-code-violation 145 duration 10
```

QoS on T3 Channels: Example

QoS on the T3 channels is supported for both PPP and HDLC encapsulation. The following example shows a typical QoS configuration for T3 interfaces:

```
class-map VOIP
match dscp EF
end-class-map
class-map OAM
match dscp AF43
end-class-map
!
Policy-map T3-no-priority
class OAM
bandwidth percent 30
!
class class-default
!
end-policy-map
!
Policy-map T3-priority
class VOIP
priority level 1
     police rate percent 60
!
class OAM
bandwidth percent 30
!
class class-default
!
end-policy-map
```

Additional References

The following sections provide references related to T3 and T1 controllers.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*

Related Topic	Document Title
Initial system bootup and configuration information for a router using Cisco IOS XR software	*Cisco IOS XR Getting Started Guide*
Cisco IOS XR AAA services configuration information	*Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
• IF-MIB • DS3-MIB • CISCO-DS3-MIB • DS1-MIB **Note** Not supported on the 4-Port Clear Channel T3/E3 SPA. • Entity MIBs	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Dense Wavelength Division Multiplexing Controllers on the Cisco ASR 9000 Series Router

This module describes the configuration of dense wavelength division multiplexing (DWDM) controllers on the Cisco ASR 9000 Series Aggregation Services Routers.

DWDM is an optical technology that is used to increase bandwidth over existing fiber-optic backbones. DWDM can be configured on supported 10-Gigabit Ethernet (GE) line cards. After you configure the DWDM controller, you can configure an associated 10-Gigabit Ethernet interface.

Feature History for Configuring DWDM Controller Interfaces

Release	Modification
Release 3.9.0	This feature was introduced on the Cisco ASR 9000 Series Router on the following cards: • Cisco 8-Port 10 Gigabit Ethernet Line Card (A9K-8T-L and -E) • Cisco 2-port 10 Gigabit Ethernet + 20-port Gigabit Ethernet Combination Line Card (A9K-2T20GE-L)
Release 3.9.1	Support for the following cards was added: • Cisco 8-Port 10 Gigabit Ethernet Line Card (A9K-8T-B) • Cisco 2-port 10 Gigabit Ethernet + 20-port Gigabit Ethernet Combination Line Card (A9K-2T20GE-B and -E)
Release 4.0.0	Support for IPoDWDM Proactive Protection was added on the following cards: • Cisco 8-Port 10 Gigabit Ethernet Line Card (A9K-8T-L, -B, and -E) • Cisco 2-port 10 Gigabit Ethernet + 20-port Gigabit Ethernet Combination Line Card (A9K-2T20GE-L, -B, and -E)

Release 4.2.1	Support for IPoDWDM Proactive Protection was added on these Module Port Adaptors: • A9K-MPA-4x10GE • A9K-MPA-2X10GE
Release 4.2.3	Support for IPoDWDM Proactive Protection was added on these Module Port Adaptors: • A9K-MPA-2X40GE • A9K-MPA-1X40GE

Contents

Prerequisites for Configuring DWDM Controller Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring a DWDM controller, be sure that you have installed one of the following cards that support DWDM:

- Cisco 8-Port 10 Gigabit Ethernet Line Card
- Cisco 2-port 10 Gigabit Ethernet + 20-port Gigabit Ethernet Combination Line Card

Information About the DWDM Controllers

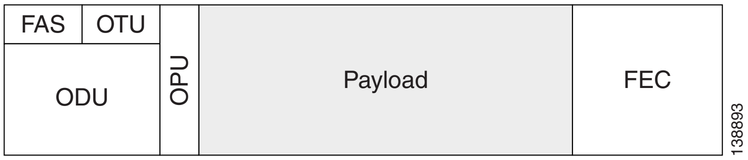
DWDM support in Cisco IOS XR software is based on the Optical Transport Network (OTN) protocol that is specified in ITU-T G.709. This standard combines the benefits of SONET/SDH technology with the multiwavelength networks of DWDM. It also provides for forward error correction (FEC) that can allow a reduction in network costs by reducing the number of regenerators used.

To enable multiservice transport, OTN uses the concept of a wrapped overhead (OH). To illustrate this structure:

- Optical channel payload unit (OPU) OH information is added to the information payload to form the OPU. The OPU OH includes information to support the adaptation of client signals.

- Optical channel data unit (ODU) OH is added to the OPU to create the ODU. The ODU OH includes information for maintenance and operational functions to support optical channels.
- Optical channel transport unit (OTU) OH together with the FEC is added to form the OTU. The OTU OH includes information for operational functions to support the transport by way of one or more optical channel connections.
- Optical channel (OCh) OH is added to form the OCh. The OCh provides the OTN management functionality and contains four subparts: the OPU, ODU, OTU, and frame alignment signal (FAS). See Figure 33.

Figure 33 OTN Optical Channel Structure



Information about IPoDWDM

Cisco IOS XR software includes the IP over Dense Wavelength Division Multiplexing (IPoDWDM) feature.

IPoDWDM is supported on the following hardware devices:

- Cisco 8-Port 10 Gigabit Ethernet Line Card
- Cisco 2-port 10 Gigabit Ethernet + 20-port Gigabit Ethernet Combination Line Card

IPoDWDM currently provides the following software features:

- Proactive Maintenance

Proactive Maintenance

Proactive maintenance automatically triggers Forward Error Correction-Fast Re-Route (FEC-FRR). Proactive maintenance requires coordinated maintenance between Layer 0 (L0) and Layer 3 (L3). L0 is the DWDM optical layer. FEC-FRR is an L3 protection mechanism. FEC-FRR detects failures before they happen and corrects errors introduced during transmission or that are due to a degrading signal.

System administrators can configure the following IPoDWDM features:

- Optical Layer DWDM port, see Configuring the Optical Layer DWDM Ports, page 473.
- Administrative state of DWDM optical ports, see Configuring the Administrative State of DWDM Optical Ports, page 475.
- FEC-FRR trigger threshold, window size, revert threshold, and revert window size, see Configuring Proactive FEC-FRR Triggering, page 477.

FEC-FRR Triggering

FEC-FRR can be configure to be triggered by the following alarms:

- ais – Alarm Indication Signal (AIS)
- bdi – Backward Defect Indication (BDI)

- *bdiO – Backward Defect Indication - Overhead (BDI-O)
- *bdiP – Backward Defect Indication - Payload (BDI-P)
- *deg – Degraded (DEG)
- lck – Locked (LCK)
- lof – Loss of Frame (LOF)
- lom – Loss of Multi Frame
- los – Loss of Signal (LOS)
- *losO – Loss of Signal - Overhead (LOS-O)
- *losP – Loss of Signal - Payload (LOS-P)
- oci – Open Connection Indication (OCI)
- plm – Payload Mismatch (PLM)
- *ssf – Server Signal Failure (SSF)
- *ssfO – Server Signal Failure - Overhead (SSF-O)
- *ssfP – Server Signal Failure - Payload (SSF-P)
- tim – Trace Identifier Mismatch (TIM)

Signal Logging

DWDM statistic data, such as EC, UC and alarms, are collected and stored in the log file on the DWDM line card.

How to Configure DWDM Controllers

The DWDM controllers are configured in the physical layer control element of the Cisco IOS XR software configuration space. This configuration is done using the **controller dwdm** command, and is described in the following task:

- Configuring G.709 Parameters, page 466

Note All interface configuration tasks for Gigabit Ethernet interfaces still must be performed in interface configuration mode.

Configuring G.709 Parameters

This task describes how to customize the alarm display and the thresholds for alerts and forward error correction (FEC). You need to use this task only if the default values are not correct for your installation.

Prerequisites

The **loopback**, and **g709 fec** commands can be used only when the controller is in the shutdown state. Use the **admin-state** command.

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *interface-path-id*
3. **admin-state maintenance**
   or
   admin-state out-of-service
4. **commit**
5. **loopback** {**internal** | **line**}
6. **g709 fec** {**disable** | **enhanced** | **standard**}
7. **g709** {**odu** | **otu**} **report** *alarm* **disable**
8. **g709 otu overhead tti** {**expected** | **sent**} {**ascii** | **hex**} *tti-string*
9. **end**
   or
   commit
10. **admin-state in-service**
11. **show controllers dwdm** *interface-path-id* **g709**

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:Router# configure`
		Enters global configuration mode.

Step 2	**`controller dwdm`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/0`
		Specifies the DWDM controller name in the notation *rack/slot/module/port* and enters DWDM configuration mode.

Step 3	**`admin-state maintenance`**
	or
	`admin-state out-of-service`
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# admin-state out-of-service`
		Disables the DWDM controller. You must disable the controller before you can use the DWDM configuration commands.

Step 4	**`commit`**
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# commit`
		Saves configuration changes. This performs the shutdown from the previous step. When the controller has been shut down, you can proceed with the configuration.

	Command or Action	Purpose

Step 5	**loopback** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# loopback internal`
		(Optional) Configures the DWDM controller for loopback mode.

Step 6	**g709 fec** {**disable** | **standard**}
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# g709 fec disable`
		(Optional) Configures the forward error correction mode (FEC) for the DWDM controller. By default, enhanced FEC is enabled.

Step 7	**g709** {**odu** | **otu**} **report** *alarm* **disable** **Example:** `RP/0/RSP0/CPU0:Router(config-dwdm)# g709 odu bdi disable`	(Optional) Disables the logging of selected optical channel data unit (ODU) alarms or optical channel transport unit (OTU) alarms to the console for a DWDM controller. By default, all alarms are logged to the console.
Step 8	**g709 otu overhead tti** {**expected** | **sent**} {**ascii** | **hex**} *tti-string* **Example:** `RP/0/RSP0/CPU0:Router(config-dwdm)# g709 otu overhead tti expected ascii test OTU 5678`	Configures a transmit or expected Trail Trace Identifier (TTI) that is displayed in the **show controller dwdm** command.

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# end`
	or
	`RP/0/RSP0/CPU0:Router(config-dwdm)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 10	**admin-state in-service** **Example:** `RP/0/RSP0/CPU0:Router(config-dwdm)# admin-state in-service`	Places the DWDM port in In Service (IS) state, to support all normal operation.
Step 11	**show controllers dwdm** *interface-path-id* **g709** **Example:** `RP/0/RSP0/CPU0:Router# show controller dwdm 0/1/0/0 optics`	Displays the G.709 Optical Transport Network (OTN) protocol alarms and counters for Bit Errors, along with the FEC statistics and threshold-based alerts.

What to Do Next

All interface configuration tasks for the Gigabit Ethernet interfaces still must be performed in interface configuration mode. Refer to the corresponding modules in this book for more information.

How to Perform Performance Monitoring on DWDM Controllers

Performance monitoring parameters are used to gather, store, set thresholds for, and report performance data for early detection of problems. Thresholds are used to set error levels for each performance monitoring parameter. During the accumulation cycle, if the current value of a performance monitoring parameter reaches or exceeds its corresponding threshold value, a threshold crossing alert (TCA) can be generated. The TCAs provide early detection of performance degradation.

Performance monitoring statistics are accumulated on a 15-minute basis, synchronized to the start of each quarter-hour. They are also accumulated on a daily basis starting at midnight. Historical counts are maintained for thirty-three 15-minute intervals and two daily intervals.

Performance monitoring is described in the following task:

- Configuring DWDM Controller Performance Monitoring, page 469

Configuring DWDM Controller Performance Monitoring

This task describes how to configure performance monitoring on DWDM controllers and how to display the performance parameters.

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *interface-path-id*
3. **pm** {**15-min** | **24-hour**} **fec threshold** {**ec-bits** | **uc-words**} *threshold*
4. **pm** {**15-min** | **24-hour**} **optics threshold** {**lbc** | **opr** | **opt**} {**max** | **min**} *threshold*
5. **pm** {**15-min** | **24-hour**} **otn threshold** *otn-parameter threshold*
6. **pm** {**15-min** | **24-hour**} **fec report** {**ec-bits** | **uc-words**} **enable**
7. **pm** {**15-min** | **24-hour**} **optics report** {**lbc** | **opr** | **opt**} {**max-tca** | **min-tca**} **enable**

8. **pm** {**15-min** | **24-hour**} **otn report** *otn-parameter* **enable**
9. **end**
   or
   commit
10. **show controllers dwdm** *interface-path-id* **pm history** [**15-min** | **24-hour** | **fec** | **optics** | **otn**]
11. **show controllers dwdm** *interface-path-id* **pm interval** {**15-min** | **24-hour**} [**fec** | **optics** | **otn**] *index*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:Router# configure`	Enters global configuration mode.
Step 2	**controller dwdm** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/0`	Specifies the DWDM controller name in the notation *rack/slot/module/port* and enters DWDM configuration mode.

Step 3	**pm** {**15-min** | **24-hour**} **fec threshold** {**ec-bits** | **uc-words**} *threshold*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min fec threshold ec-bits 49000000`
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min fec threshold uc-words xxxxxx`
		Configures a performance monitoring threshold for specific parameters on the FEC layer.

Step 4	**pm** {**15-min** | **24-hour**} **optics threshold** {**lbc** | **opr** | **opt**} {**max** | **min**} *threshold*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold opt max xxx`
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold lbc min xxx`
		Configures a performance monitoring threshold for specific parameters on the optics layer.

	Command or Action	Purpose

Step 5	**pm** {**15-min** | **24-hour**} **otn threshold** *otn-parameter threshold*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min otn threshold bbe-pm-ne xxx`
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min otn threshold es-sm-fe xxx`
		Configures a performance monitoring threshold for specific parameters on the optical transport network (OTN) layer. OTN parameters can be as follows:
		• **bbe-pm-fe**—Far-end path monitoring background block errors (BBE-PM)
		• **bbe-pm-ne**—Near-end path monitoring background block errors (BBE-PM)
		• **bbe-sm-fe**—Far-end section monitoring background block errors (BBE-SM)
		• **bbe-sm-ne**—Near-end section monitoring background block errors (BBE-SM)
		• **bber-pm-fe**—Far-end path monitoring background block errors ratio (BBER-PM)
		• **bber-pm-ne**—Near-end path monitoring background block errors ratio (BBER-PM)
		• **bber-sm-fe**—Far-end section monitoring background block errors ratio (BBER-SM)
		• **bber-sm-ne**—Near-end section monitoring background block errors ratio (BBER-SM)
		• **es-pm-fe**—Far-end path monitoring errored seconds (ES-PM)
		• **es-pm-ne**—Near-end path monitoring errored seconds (ES-PM)
		• **es-sm-fe**—Far-end section monitoring errored seconds (ES-SM)
		• **es-sm-ne**—Near-end section monitoring errored seconds (ES-SM)
		• **esr-pm-fe**—Far-end path monitoring errored seconds ratio (ESR-PM)
		• **esr-pm-ne**—Near-end path monitoring errored seconds ratio (ESR-PM)
		• **esr-sm-fe**—Far-end section monitoring errored seconds ratio (ESR-SM)
		• **esr-sm-ne**—Near-end section monitoring errored seconds ratio (ESR-SM)
		• **fc-pm-fe**—Far-end path monitoring failure counts (FC-PM)
		• **fc-pm-ne**—Near-end path monitoring failure counts (FC-PM)
		• **fc-sm-fe**—Far-end section monitoring failure counts (FC-SM)
		• **fc-sm-ne**—Near-end section monitoring failure counts (FC-SM)

	Command or Action	Purpose
		• **ses-pm-fe**—Far-end path monitoring severely errored seconds (SES-PM) • **ses-pm-ne**—Near-end path monitoring severely errored seconds (SES-PM) • **ses-sm-fe**—Far-end section monitoring severely errored seconds (SES-SM) • **ses-sm-ne**—Near-end section monitoring severely errored seconds (SES-SM) • **sesr-pm-fe**—Far-end path monitoring severely errored seconds ratio (SESR-PM) • **sesr-pm-ne**—Near-end path monitoring severely errored seconds ratio (SESR-PM) • **sesr-sm-fe**—Far-end section monitoring severely errored seconds ratio (SESR-SM) • **sesr-sm-ne**—Near-end section monitoring severely errored seconds ratio (SESR-SM) • **uas-pm-fe**—Far-end path monitoring unavailable seconds (UAS-PM) • **uas-pm-ne**—Near-end path monitoring unavailable seconds (UAS-PM) • **uas-sm-fe**—Far-end section monitoring unavailable seconds (UAS-SM) • **uas-sm-ne**—Near-end section monitoring unavailable seconds (UAS-SM)

Step 6	**pm** {**15-min** | **24-hour**} **fec report** {**ec-bits** | **uc-words**} **enable**
	
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min fec report ec-bits enable`
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min fec report uc-words enable`
		Configures threshold crossing alert (TCA) generation for specific parameters on the FEC layer.

Step 7	**pm** {**15-min** | **24-hour**} **optics report** {**lbc** | **opr** | **opt**} {**max-tca** | **min-tca**} **enable**
	
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report opt enable`
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report lbc enable`
		Configures TCA generation for specific parameters on the optics layer.

	Command or Action	Purpose

Step 8	**pm** {**15-min** | **24-hour**} **otn report** *otn-parameter* **enable**
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min otn report bbe-pm-ne enable`
	`RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min otn report es-sm-fe enable`
		Configures TCA generation for specific parameters on the optical transport network (OTN) layer. OTN parameters are shown in Step 5.

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# end`
	or
	`RP/0/RSP0/CPU0:Router(config-dwdm)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring IPoDWDM

This section provides the following configuration procedures:

- Configuring the Optical Layer DWDM Ports, page 473
- Configuring the Administrative State of DWDM Optical Ports, page 475
- Configuring Proactive FEC-FRR Triggering, page 477

Configuring the Optical Layer DWDM Ports

Use the following procedure to configure the Optical Layer DWDM ports.

SUMMARY STEPS

1. **configure**

2. **controller dwdm** *interface-path-id*
3. **network port id** *id-number*
4. **network connection id** *id-number*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:Router# config`
		Enters global configuration mode.

Step 2	**`controller dwdm`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/1`
		Specifies the DWDM controller and enters DWDM controller mode.

Step 3	**`network port id`** *`id-number`*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# network port id 1/0/1/1`
		Assigns an identifier number to a port for the Multi Service Transport Protocol (MSTP).

	Command or Action	Purpose
Step 4	`network connection id` *id-number* **Example:** `RP/0/RSP0/CPU0:Router(config-dwdm)# network connection id 1/1/1/1`	Configures a connection identifier for the Multi Service Transport Protocol (MSTP).

Step 5	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# end`
	or
	`RP/0/RSP0/CPU0:Router(config-dwdm)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Administrative State of DWDM Optical Ports

Use the following procedure to configure the administrative state and optionally set the maintenance embargo flag.

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *interface-path-id*
3. **admin-state** {**in-service** | **maintenance** | **out-of-service**}
4. **exit**
5. **interface tengige** *interface-path-id*
6. **maintenance disable**
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:Router# config`
		Enters global configuration mode.

Step 2	**`controller dwdm`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:Routerconfig)# controller dwdm 0/1/0/1`
		Specifies the DWDM controller and enters DWDM controller mode.

Step 3	**`admin-state`** `{`**`in-service`** `|` **`maintenance`** `|` **`out-of-service`**`}`
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# admin-state maintenance`
		Specifies the transport administration state.

Step 4	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# exit`
		Exits to the previous mode.

Step 5	**`interface pos`** *`interface-path-id`*
	or
	`interface tengige` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:Router(config)# interface pos 1/0/1/1`
	or
	`RP/0/RSP0/CPU0:Router(config)# interface tengige 1/0/1/1`
		Specifies the interface and enters interface configuration mode.

	Command or Action	Purpose
Step 6	**maintenance disable** **Example:** `RP/0/RSP0/CPU0:Router(config-if)# maintenance disable`	Provisions the maintenance embargo flag, which prevents maintenance activities from being performed on an interface.

Step 7	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# end`
	or
	`RP/0/RSP0/CPU0:Router(config-dwdm)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Proactive FEC-FRR Triggering

Use the following procedure to configure automatic triggering of Forward Error Correction-Fast Re-Route (FEC-FRR).

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *interface-path-id*
3. **proactive**
4. **logging signal** *file-name*
5. **proactive trigger threshold** *x-coefficient y-power*
6. **proactive trigger window** *window*
7. **proactive revert threshold** *x-coefficient y-power*
8. **proactive revert window** *window*
9. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:Router# config`
		Enters global configuration mode.

Step 2	**`controller dwdm`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/1`
		Specifies the DWDM controller and enters DWDM controller mode.

Step 3	**`proactive`**
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# proactive enable`
		Enables automatic triggering of FEC-FRR.

Step 4	**`logging signal`** *`file-name`*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# logging signal LogFile1`
		Enables10 millisecond proactive monitoring of FEC-FRR.

Step 5	**`proactive trigger threshold`** *`x-coefficient y-power`*
	Example:
	`RP/0/RSP0/CPU0:Routerconfig-dwdm)# proactive trigger threshold 1 9`
		Configures the trigger threshold of FEC-FRR in the form of *x*E-*y*.

Step 6	**`proactive trigger window`** *`window`*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# proactive trigger window 10000`
		Configures the trigger window (in milliseconds) in which FRR may be triggered.

Step 7	**`proactive revert threshold`** *`x-coefficient y-power`*
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# proactive revert threshold 1 9`
		Configures the revert threshold (in the form of *x*E-*y*) to trigger reverting from the FEC-FRR route back to the original route.

	Command or Action	Purpose
Step 8	**proactive revert window** *window* **Example:** `RP/0/RSP0/CPU0:Router(config-dwdm)# proactive revert window 600000`	Configures the revert window in which reverting from the FEC-FRR route back to the original route is triggered.

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:Router(config-dwdm)# end`
	or
	`RP/0/RSP0/CPU0:Router(config-dwdm)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuration Examples

This section includes the following examples:

- Turning On the Laser: Example, page 479
- Turning Off the Laser: Example, page 480
- DWDM Controller Configuration: Examples, page 480
- DWDM Performance Monitoring: Examples, page 480
- IPoDWDM Configuration: Examples, page 481

Turning On the Laser: Example

Note This is a required configuration. The DWDM cards will not operate without this configuration.

The following example shows how to turn on the laser and place a DWDM port in In Service (IS) state:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:Router(config)# controller dwdm 0/1/0/1
RP/0/RP0/CPU0:Router(config-dwdm)# admin-state in-service
RP/0/RP0/CPU0:Router(config-dwdm)# commit
```

Turning Off the Laser: Example

The following example shows how to turn off the laser, stop all traffic and place a DWDM port in Out of Service (OOS) state:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:Router(config)# controller dwdm 0/1/0/1
RP/0/RP0/CPU0:Router(config-dwdm)# admin-state out-of-service
RP/0/RP0/CPU0:Router(config-dwdm)# commit
```

DWDM Controller Configuration: Examples

The following example shows how to customize the alarm display and the thresholds for alerts and forward error correction (FEC):

```
RP/0/RSP0/CPU0:Router# configure
RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/0
RP/0/RSP0/CPU0:Router(config-dwdm)# maintenance out-of-service
RP/0/RSP0/CPU0:Router(config-dwdm)# commit
RP/0/RSP0/CPU0:Router(config-dwdm)# g709 disable
RP/0/RSP0/CPU0:Router(config-dwdm)# loopback internal
RP/0/RSP0/CPU0:Router(config-dwdm)# g709 fec standard
RP/0/RSP0/CPU0:Router(config-dwdm)# g709 odu bdi disable
RP/0/RSP0/CPU0:Router(config-dwdm)# maintenance in-service
RP/0/RSP0/CPU0:Router(config-dwdm)# commit
```

DWDM Performance Monitoring: Examples

The following example shows how to configure performance monitoring for the optics parameters and how to display the configuration and current statistics:

```
RP/0/RSP0/CPU0:Router# configure
RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/2/0/0

RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold opt max 2000000
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold opt min 200
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold lbc max 3000000
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold lbc min 300
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold opr max 4000000
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics threshold opr min 400
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report opt max-tca enable
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report opt min-tca enable
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report opr max-tca enable
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report opr min-tca enable
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report lbc max-tca enable
RP/0/RSP0/CPU0:Router(config-dwdm)# pm 15-min optics report lbc min-tca enable
RP/0/RSP0/CPU0:Router(config-dwdm)# exit
RP/0/RSP0/CPU0:Router(config)# exit

Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:y

LC/0/2/CPU0:Jul 12 04:10:47.252 : plim_4p_10ge_dwdm[194]: %L1-PMENGINE-4-TCA : Port DWDM
0/2/0/0 reports OPTICS TX-PWR-MIN(NE) PM TCA with current value 0, threshold 200 in
current 15-min interval window
LC/0/2/CPU0:Jul 12 04:10:47.255 : plim_4p_10ge_dwdm[194]: %L1-PMENGINE-4-TCA : Port DWDM
0/2/0/0 reports OPTICS RX-PWR-MIN(NE) PM TCA with current value 68, threshold 400 in
current 15-min interval window
```

```
RP/0/RP1/CPU0:Jul 12 04:09:05.443 : config[65678]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'lab'. Use 'show configuration commit changes 1000000001'
to view the changes.
RP/0/RP1/CPU0:Jul 12 04:09:05.604 : config[65678]: %MGBL-SYS-5-CONFIG_I : Configured from
console by lab

RP/0/RSP0/CPU0:Router# show controllers dwdm 0/2/0/0 pm interval 15-min optics 0

Optics in the current interval [ 4:15:00 - 04:26:02 Wed Jul 12 2006]
             MIN     AVG     MAX  Threshold  TCA  Threshold  TCA
                                   (min)  (enable) (max)  (enable)
LBC[mA ] :  3605    4948    6453    300        YES     3000000   YES
OPT[uW]  :  2593    2593    2593    200        YES     2000000   YES
OPR[uW]  :  69      69      70      400        YES     4000000   YES
```

IPoDWDM Configuration: Examples

This section includes the following examples:

- Optical Layer DWDM Port Configuration: Examples, page 481
- Administrative State of DWDM Optical Ports Configuration: Examples, page 481
- Proactive FEC-FRR Triggering Configuration: Examples, page 482

Optical Layer DWDM Port Configuration: Examples

The following example shows how to configure Optical Layer DWDM ports.

```
RP/0/RSP0/CPU0:Router# configure
RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/1
RP/0/RSP0/CPU0:Router(config-dwdm)# network port id 1/0/1/1
RP/0/RSP0/CPU0:Router(config-dwdm)# network connection id 1/1/1/1
```

Administrative State of DWDM Optical Ports Configuration: Examples

The following examples show how to configure the administrative state and optionally set the maintenance embargo flag:

```
RP/0/RSP0/CPU0:Router# configure
RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/1
RP/0/RSP0/CPU0:Router(config-dwdm)# admin-state in-service
RP/0/RSP0/CPU0:Router(config-dwdm)# exit
RP/0/RSP0/CPU0:Router(config)# interface tengige 1/0/1/1
RP/0/RSP0/CPU0:Router(config-if)# maintenance disable
RP/0/RSP0/CPU0:Router(config-if)# commit
```

Proactive FEC-FRR Triggering Configuration: Examples

This example shows how to configure automatic triggering of Forward Error Correction-Fast Re-Route (FEC-FRR):

```
RP/0/RSP0/CPU0:Router# configure
RP/0/RSP0/CPU0:Router(config)# controller dwdm 0/1/0/1
RP/0/RSP0/CPU0:Router(config-dwdm)# proactive
RP/0/RSP0/CPU0:Router(config-dwdm)# logging signal LogFile1
RP/0/RSP0/CPU0:Router(config-dwdm)# proactive trigger threshold 1 9
RP/0/RSP0/CPU0:Router(config-dwdm)# proactive trigger window 10000
RP/0/RSP0/CPU0:Router(config-dwdm)# proactive revert threshold 1 9
RP/0/RSP0/CPU0:Router(config-dwdm)# proactive revert window 600000
```

Additional References

The following sections provide references related to DWDM controller configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using Cisco IOS XR software	*Cisco IOS XR Getting Started Guide*
Cisco IOS XR AAA services configuration information	*Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference*

Standards

Standards	Title
ITU-T G.709/Y.1331	Interfaces for the optical transport network (OTN)

MIBs

MIBs	MIBs Link
—	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml
OTN-MIB	IPoDWDM MIB

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring POS Interfaces onthe Cisco ASR 9000 Series Router

This module describes the configuration of Packet-over-SONET/SDH (POS) interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

POS interfaces provide secure and reliable data transmission over SONET and Synchronous Digital Hierarchy (SDH) frames using Cisco High-Level Data Link Control (HDLC) protocol or Point-to-Point Protocol (PPP) encapsulation. In addition to Cisco HDLC and PPP encapsulation, the Cisco ASR 9000 Series Router supports Frame Relay encapsulation.

The commands for configuring Layer 1 POS interfaces are provided in the *Cisco IOS XR Interface and Hardware Component Command Reference*.

Feature History for Configuring POS Interfaces on Cisco IOS XR Software

Release	Modification
Release 4.0.0	This feature was introduced on the Cisco ASR 9000 Series Router on the following SPAs: • Cisco 1-Port Channelized OC-48/STM-16 SPA • Cisco 2-Port Channelized OC-12c/DS0 SPA • Cisco 1-Port OC-192c/STM-64 POS/RPR XFP SPA • Cisco 2-Port OC-48c/STM-16 POS/RPR SPA • Cisco 8-Port OC-12c/STM-4 POS SPA
Release 4.0.1	Support for the following SPAs was added on the Cisco ASR 9000 Series Router: • Cisco 4-Port OC-3c/STM-1 POS SPA • Cisco 8-Port OC-3c/STM-1 POS SPA

Contents

Prerequisites for Configuring POS Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring POS interfaces, be sure that the following conditions are met:

- You know the IP address of the interface you will assign to the new POS interface configuration.
- You have configured a clear channel or channelized SONET controller, as described in the "Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router" or "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router" modules.

Information About Configuring POS Interfaces

To configure POS interfaces, you must understand the following concepts:

- Cisco HDLC Encapsulation, page 487
- PPP Encapsulation, page 487
- Keepalive Timer, page 488
- Frame Relay Encapsulation, page 489
- Default Settings for POS Interfaces, page 486

On the Cisco ASR 9000 Series Router, a single POS interface carries data using PPP, Cisco HDLC, or Frame Relay encapsulation.

The router identifies the POS interface address by the physical layer interface module (PLIM) card rack number, slot number, bay number, and port number that are associated with that interface. If a subinterface and permanent virtual circuits (PVCs) are configured under the POS interface, then the router includes the subinterface number in the POS interface path ID.

Default Settings for POS Interfaces

When a POS interface is brought up and no additional configuration commands are applied, the default interface settings shown in Table 13 are present. These default settings can be changed by configuration.

Table 13 POS Modular Services Card and PLIM Default Interface Settings

Parameter	Configuration File Entry	Default Settings
Keepalive **Note** The **keepalive** command applies to POS interfaces using HDLC or PPP encapsulation. It does not apply to POS interfaces using Frame Relay encapsulation.	**keepalive** {*interval* [*retry*] | **disable**} **no keepalive**	Interval of 10 seconds Retry of: • 5 (with PPP encapsulation) • 3 (with HDLC encapsulation)
Encapsulation	**encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]]	**hdlc**
Maximum transmission unit (MTU)	**mtu** *bytes*	4474 bytes
Cyclic redundancy check (CRC)	**crc** [**16** | **32**]	**32**

Note Default settings do not appear in the output of the **show running-config** command.

Cisco HDLC Encapsulation

Cisco High-Level Data Link Controller (HDLC) is the Cisco proprietary protocol for sending data over synchronous serial links using HDLC. Cisco HDLC also provides a simple control protocol called Serial Line Address Resolution Protocol (SLARP) to maintain serial link keepalives. HDLC is the default encapsulation type for POS interfaces under Cisco IOS XR software. Cisco HDLC is the default for data encapsulation at Layer 2 (data link) of the Open System Interconnection (OSI) stack for efficient packet delineation and error control.

Note Cisco HDLC is enabled by default for POS interfaces.

Cisco HDLC uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 488.

PPP Encapsulation

PPP is a standard protocol used to send data over synchronous serial links. PPP also provides a Link Control Protocol (LCP) for negotiating properties of the link. LCP uses echo requests and responses to monitor the continuing availability of the link.

Note When an interface is configured with PPP encapsulation, a link is declared down, and full LCP negotiation is re-initiated after three ECHOREQ packets are sent without receiving an ECHOREP response.

PPP provides the following Network Control Protocols (NCPs) for negotiating the properties of data protocols that run on the link:

- IP Control Protocol (IPCP)—negotiates IP properties
- Multiprotocol Label Switching control processor (MPLSCP)—negotiates MPLS properties
- Cisco Discovery Protocol control processor (CDPCP)—negotiates CDP properties
- IPv6CP—negotiates IP Version 6 (IPv6) properties
- Open Systems Interconnection control processor (OSICP)—negotiates OSI properties

PPP uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 488.

PPP supports the following authentication protocols, which require a remote device to prove its identity before allowing data traffic to flow over a connection:

- Challenge Handshake Authentication Protocol (CHAP)—CHAP authentication sends a challenge message to the remote device. The remote device encrypts the challenge value with a shared secret and returns the encrypted value and its name to the local router in a response message. The local router attempts to match the remote device's name with an associated secret stored in the local username or remote security server database; it uses the stored secret to encrypt the original challenge and verify that the encrypted values match.
- Microsoft Challenge Handshake Authentication Protocol (MS-CHAP)—MS-CHAP is the Microsoft version of CHAP. Like the standard version of CHAP, MS-CHAP is used for PPP authentication; in this case, authentication occurs between a personal computer using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server.
- Password Authentication Protocol (PAP)—PAP authentication requires the remote device to send a name and a password, which are checked against a matching entry in the local username database or in the remote security server database.

Note For more information on enabling and configuring PPP authentication protocols, see the "Configuring PPP on the Cisco ASR 9000 Series Router" module later in this manual.

Use the **ppp authentication** command in interface configuration mode to enable CHAP, MS-CHAP, and PAP on a POS interface.

Note Enabling or disabling PPP authentication does not effect the local router's willingness to authenticate itself to the remote device.

Keepalive Timer

Cisco keepalives are useful for monitoring the link state. Periodic keepalives are sent to and received from the peer at a frequency determined by the value of the keepalive timer. If an acceptable keepalive response is not received from the peer, the link makes the transition to the down state. As soon as an acceptable keepalive response is obtained from the peer or if keepalives are disabled, the link makes the transition to the up state.

If three keepalives are sent to the peer and no response is received from peer, then the link makes the transition to the down state. ECHOREQ packets are sent out only when LCP negotiation is complete (for example, when LCP is open).

Note The **keepalive** command applies to POS interfaces using HDLC or PPP encapsulation. It does not apply to POS interfaces using Frame Relay encapsulation.

Use the **keepalive** command in interface configuration mode to set the frequency at which LCP sends ECHOREQ packets to its peer. To restore the system to the default keepalive interval of 10 seconds, use the **keepalive** command with **no** argument. To disable keepalives, use the **keepalive disable** command. For both PPP and Cisco HDLC, a keepalive of 0 disables keepalives and is reported in the **show running-config** command output as **keepalive disable**.

To remove the **keepalive** command from the configuration entirely, use the **no keepalive** command. You must remove the **keepalive** command from an interface configuration before you can configure Frame Relay encapsulation on that interface. Frame Relay interfaces do not support keepalives.

Note During MDR, the keepalive interval must be 10 seconds or more.

When LCP is running on the peer and receives an ECHOREQ packet, it responds with an echo reply (ECHOREP) packet, regardless of whether keepalives are enabled on the peer.

Keepalives are independent between the two peers. One peer end can have keepalives enabled while the other end has them disabled. Even if keepalives are disabled locally, LCP still responds with ECHOREP packets to the ECHOREQ packets it receives. Similarly, LCP also works if the period of keepalives at each end is different.

Note Use the **debug chdlc slarp packet** command and other Cisco HDLC **debug** commands to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

Frame Relay Encapsulation

On the Cisco ASR 9000 Series Router, Frame Relay encapsulated POS interface configuration is hierarchical and comprises the following elements:

1. The POS main interface is comprised of the physical interface and port. If you are not using the POS interface to support Cisco HDLC and PPP encapsulated connections, then you must configure subinterfaces with PVCs under the POS main interface. Frame Relay connections are supported on PVCs only.
2. POS subinterfaces are configured under the POS main interface. A POS subinterface does not actively carry traffic until you configure a PVC under the POS subinterface.
3. Point-to-point and Layer 2 attachment circut (AC) PVCs are configured under a POS subinterface. You cannot configure a PVC directly under a main interface. A single point-to-point or L2 AC PVC is allowed per subinterface. PVCs use a predefined circuit path and fail if the path is interrupted. PVCs remain active until the circuit is removed. Connections on the POS PVC support Frame Relay encapsulation only.
4. Layer 3 configuration typically takes place on the subinterface.

Note The administrative state of a parent interface drives the state of the subinterface and its PVC. When the administrative state of a parent interface or subinterface changes, so does the administrative state of any child PVC configured under that parent interface or subinterface.

On the Cisco ASR 9000 Series Router, the following SPAs support Frame Relay encapsulation:

- Cisco 4-Port OC-3c/STM-1 POS SPA
- Cisco 8-Port OC-3c/STM-1 POS SPA
- Cisco 1-Port OC-192c/STM-64 POS/RPR XFP SPA
- Cisco 2-Port OC-48c/STM-16 POS/RPR SPA
- Cisco 8-Port OC-12c/STM-4 POS SPA

To configure Frame Relay encapsulation on POS interfaces, use the **encapsulation frame-relay** command.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (this is the default)
- IETF

Use the **encap** command in PVC configuration mode to configure Cisco or IETF encapsulation on a PVC. If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main POS interface.

Note Cisco encapsulation is required on POS main interfaces that are configured for MPLS. IETF encapsulation is not supported for MPLS.

Before you configure Frame Relay encapsulation on an interface, you must verify that all prior Layer 3 configuration is removed from that interface. For example, you must ensure that there is no IP address configured directly under the main interface; otherwise, any Frame Relay configuration done under the main interface will not be viable.

LMI on Frame Relay Interfaces

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay UNI interface. By default, **cisco** LMI is enabled on all PVCs. However, you can modify the default LMI type to be ANSI or Q.933, as described in the "Modifying the Default Frame Relay Configuration on an Interface" module later in this manual.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA:

```
(MTU - 13)/8 = maximum number of PVCs
```

Note The default setting of the **mtu** command for a POS interface is 4474 bytes. Therefore, the default numbers of PVCs supported on a POS interface configured with **cisco** LMI is 557.

Note You must configure the LMI interface type on Frame Relay interfaces; otherwise, the POS interface does not come up. For connections between Provider Edge (PE) and Customer Edge (CE) routers, the PE end must be DCE and the CE end must be DTE for LMI to come up. For more information about configuring the LMI interface type on a Frame Relay interface, see the "Configuring Frame Relay on the Cisco ASR 9000 Series Router" module.

How to Configure a POS Interface

This section contains the following procedures:

- Bringing Up a POS Interface, page 491
- Configuring Optional POS Interface Parameters, page 494
- Creating a Point-to-Point POS Subinterface with a PVC, page 496
- Configuring Optional PVC Parameters, page 498
- Modifying the Keepalive Interval on POS Interfaces, page 501
- Creating a Layer 2 Frame Relay Subinterface with a PVC, page 504

Bringing Up a POS Interface

This task describes the commands you can use to bring up a POS interface.

Prerequisites

You must have a POS line card or SPA installed in a router that is running Cisco IOS XR software.

Restrictions

The configuration on both ends of the POS connection must match for the interface to be active.

SUMMARY STEPS

1. **show interfaces**
2. **configure**
3. **interface pos** *interface-path-id*
4. **ipv4 address** *ipv4_address/prefix*
5. **no shutdown**
6. **end**
   or
   commit
7. **exit**
8. **exit**

9. Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
10. **show ipv4 interface brief**
11. **show interfaces pos** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show interfaces** **Example:** `RP/0/RSP0/CPU0:router# show interfaces`	(Optional) Displays configured interfaces. • Use this command to also confirm that the router recognizes the PLIM card.
Step 2	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**interface pos** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0`	Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 4	**ipv4 address** *ipv4_address/prefix* **Example:** `RP/0/RSP0/CPU0:router (config)#ipv4 address 10.46.8.6/24`	Assigns an IP address and subnet mask to the interface. **Note** Skip this step if you are configuring Frame Relay encapsulation on this interface. For Frame Relay, the IP address and subnet mask are configured under the subinterface.
Step 5	**no shutdown** **Example:** `RP/0/RSP0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming the parent SONET layer is not configured administratively down).

	Command or Action	Purpose

Step 6	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router (config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 7	**`exit`**
	
	Example:
	`RP/0/RSP0/CPU0:router (config-if)# exit`
		Exits interface configuration mode and enters global configuration mode.

Step 8	**`exit`**
	
	Example:
	`RP/0/RSP0/CPU0:router (config)# exit`
		Exits global configuration mode and enters EXEC mode.

Step 9	**`show interfaces`**
	`configure`
	`interface pos` *`interface-path-id`*
	`no shut`
	`exit`
	`exit`
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router# show interfaces`
	`RP/0/RSP0/CPU0:router# configure`
	`RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/0`
	`RP/0/RSP0/CPU0:router (config-if)# no shutdown`
	`RP/0/RSP0/CPU0:router (config-if)# commit`
	`RP/0/RSP0/CPU0:router (config-if)# exit`
	`RP/0/RSP0/CPU0:router (config)# exit`
		Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
		Note The configuration on both ends of the POS connection must match.

	Command or Action	Purpose

Step 10	`show ipv4 interface brief`
	Example:
	`RP/0/RSP0/CPU0:router # show ipv4 interface brief`
		Verifies that the interface is active and properly configured.
		If you have brought up a POS interface properly, the "Status" field for that interface in the **show ipv4 interface brief** command output shows "Up."

Step 11	**`show interfaces pos`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router# show interfaces pos 0/3/0/0`
		(Optional) Displays the interface configuration.

What to Do Next

To modify the default configuration of the POS interface you just brought up, see the "Configuring Optional POS Interface Parameters" section on page 494.

Configuring Optional POS Interface Parameters

This task describes the commands you can use to modify the default configuration on a POS interface.

Prerequisites

Before you modify the default POS interface configuration, you must bring up the POS interface and remove the shutdown configuration, as described in the "Bringing Up a POS Interface" section on page 491.

Restrictions

The configuration on both ends of the POS connection must match for the interface to be active.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id*
3. **encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]]
4. **pos crc** {**16** | **32**}
5. **mtu** *value*
6. **end**
   or
   commit
7. **exit**
8. **exit**
9. **show interfaces pos** [*interface-path-id*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0`	Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 3	**encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]] **Example:** `RP/0/RSP0/CPU0:router(config-if)# encapsulation hdlc`	(Optional) Configures the interface encapsulation parameters and details such as HDLC or PPP. **Note** The default encapsulation is **hdlc**.

Step 4	**pos crc** {**16** | **32**}
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# pos crc 32`
		(Optional) Configures the CRC value for the interface. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode.
		Note The default CRC is **32**.

Step 5	**mtu** *value* **Example:** `RP/0/RSP0/CPU0:router(config-if)# mtu 4474`	(Optional) Configures the MTU value. • The default value is 4474. • The POS MTU range is 64–9216.

Step 6	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router (config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 7	**exit** **Example:** `RP/0/RSP0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/RSP0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 9	**show interfaces pos** [*interface-path-id*] **Example:** `RP/0/RSP0/CPU0:router# show interface pos 0/3/0/0`	(Optional) Displays general information for the specified POS interface.

What to Do Next

- To create a point-to-point Frame Relay subinterface with a PVC on the POS interface you just brought up, see the "Creating a Point-to-Point POS Subinterface with a PVC" section on page 496.
- To configure PPP authentication on POS interfaces where PPP encapsulation is enabled, see the *Configuring PPP on the Cisco ASR 9000 Series Router* module later in this manual.
- To modify the keepalive interval on POS interfaces that have Cisco HDLC or PPP encapsulation enabled, see the "Modifying the Keepalive Interval on POS Interfaces" section on page 501.
- To modify the default Frame Relay configuration on POS interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" of the *Configuring Frame Relay on the Cisco ASR 9000 Series Router* module in this manual.

Creating a Point-to-Point POS Subinterface with a PVC

The procedure in this section creates a point-to-point POS subinterface and configures a permanent virtual circuit (PVC) on that POS subinterface.

Note Subinterface and PVC creation is supported on interfaces with Frame Relay encapsulation only.

Prerequisites

Before you can create a subinterface on a POS interface, you must bring up the main POS interface with Frame Relay encapsulation, as described in the "Bringing Up a POS Interface" section on page 491.

Restrictions

Only one PVC can be configured for each point-to-point POS subinterface.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *dlci*
5. **end**
   or
   commit
6. Repeat Step 1 through Step 5 to bring up the POS subinterface and any associated PVC at the other end of the connection.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *interface-path-id.subinterface* **point-to-point** **Example:** `RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/0.1 point-to-point`	Enters POS subinterface configuration mode. Replace *subinterface* with a subinterface ID, in the range from 1 through 4294967295.
Step 3	**ipv4 address** *ipv4_address/prefix* **Example:** `RP/0/RSP0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24`	Assigns an IP address and subnet mask to the subinterface.
Step 4	**pvc** *dlci* **Example:** `RP/0/RSP0/CPU0:router (config-subif)# pvc 20`	Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode. Replace *dlci* with a PVC identifier, in the range from 16 to 1007. **Note** Only one PVC is allowed per subinterface.

	Command or Action	Purpose

Step 5	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router (config-fr-vc)# end`
	or
	`RP/0/RSP0/CPU0:router(config-fr-vc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**configure**
	interface pos *interface-path-id.subinterface*
	pvc *dlci*
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router# configure`
	`RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/1.1`
	`RP/0/RSP0/CPU0:router (config-subif)#ipv4 address 10.46.8.5/24`
	`RP/0/RSP0/CPU0:router (config-subif)# pvc 20`
	`RP/0/RSP0/CPU0:router (config-fr-vc)# commit`
		Repeat Step 1 through Step 5 to bring up the POS subinterface and any associated PVC at the other end of the connection.
		Note The DLCI (or PVC identifier) must match on both ends of the subinterface connection.
		Note When assigning an IP address and subnet mask to the subinterface at the other end of the connection, keep in mind that the addresses at both ends of the connection must be in the same subnet.

What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional PVC Parameters" section on page 498.
- To modify the default Frame Relay configuration on POS interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" of the *"Configuring Frame Relay on the Cisco ASR 9000 Series Router"* module.
- To attach a Layer 3 QOS service policy to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.

Configuring Optional PVC Parameters

This task describes the commands you can use to modify the default configuration on a POS PVC.

Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a POS subinterface, as described in the "Creating a Point-to-Point POS Subinterface with a PVC" section on page 496.

Restrictions

- The DLCI (or PVC identifier) must match on both ends of the PVC for the connection to be active.
- To change the PVC DLCI, you must delete the PVC and then add it back with the new DLCI.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface*
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end**
   or
   commit
7. Repeat Step 1 through Step 6 to configure the PVC at the other end of the connection.
8. **show frame-relay pvc** *dlci-number*
9. **show policy-map interface pos** *interface-path-id.subinterface* {**input** | **output**}
   or
   show policy-map type qos interface pos *interface-path-id.subinterface* {**input** | **output**}

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`interface pos interface-path-id.subinterface`
	Example:
	`RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/0.1`
		Enters POS subinterface configuration mode.

Step 3	`pvc dlci`
	Example:
	`RP/0/RSP0/CPU0:router (config-subif)# pvc 20`
		Enters subinterface configuration mode for the PVC.
		Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007.

	Command or Action	Purpose

Step 4	**`encap`** `[`**`cisco`** `|` **`ietf`**`]`
	Example:
	`RP/0/RSP0/CPU0:router (config-fr-vc)# encap ietf`
		(Optional) Configures the encapsulation for a Frame Relay PVC.
		Note If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main POS interface.

Step 5	**`service-policy`** `{`**`input`** `|` **`output`**`}` *`policy-map`*
	Example:
	`RP/0/RSP0/CPU0:router (config-fr-vc)# service-policy output policy1`
		Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.
		Note For information on creating and configuring policy maps, refer to the *Cisco IOS XR Modular Quality of Service Configuration Guide*,

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router (config-fr-vc)# end`
	or
	`RP/0/RSP0/CPU0:router(config-fr-vc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 7	**`configure`**
	`interface pos` *`interface-path-id.subinterface`*
	`pvc` *`dlci`*
	`encap` `[`**`cisco`** `|` **`ietf`**`]`
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
	`RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/1.1`
	`RP/0/RSP0/CPU0:router (config-subif)# pvc 20`
	`RP/0/RSP0/CPU0:router (config-fr-vc)# encap cisco`
	`RP/0/RSP0/CPU0:router (config-fr-vc)# commit`
		Repeat Step 1 through Step 6 to bring up the POS subinterface and any associated PVC at the other end of the connection.
		Note The configuration on both ends of the subinterface connection must match.

	Command or Action	Purpose
Step 8	**show frame-relay pvc** *dlci-number* **Example:** `RP/0/RSP0/CPU0:router# show frame-relay pvc 20`	(Optional) Verifies the configuration of specified POS interface.

Step 9	**show policy-map interface pos** *interface-path-id.subinterface* {**input** | **output**}
	or
	show policy-map type qos interface pos *interface-path-id.subinterface* {**input** | **output**}
	Example:
	`RP/0/RSP0/CPU0:router# show policy-map interface pos 0/3/0/0.1 output`
	or
	`RP/0/RSP0/CPU0:router# show policy-map type qos interface pos 0/3/0/0.1 output`
		(Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface.

What to Do Next

To modify the default Frame Relay configuration on POS interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" of the *"Configuring Frame Relay on the Cisco ASR 9000 Series Router"* module.

Modifying the Keepalive Interval on POS Interfaces

Perform this task to modify the keepalive interval on POS interfaces that have Cisco HDLC or PPP encapsulation enabled.

Note When you enable Cisco HDLC or PPP encapsulation on a POS interface, the default keepalive interval is 10 seconds. Use this procedure to modify that default keepalive interval.

Note Cisco HDLC is enabled by default on POS interfaces.

Prerequisites

Before you can modify the keepalive timer configuration, you must ensure that Cisco HDLC or PPP encapsulation is enabled on the interface. Use the **encapsulation** command to enable Cisco HDLC or PPP encapsulation on the interface, as described in the "Configuring Optional POS Interface Parameters" section on page 494.

Restrictions

During MDR, the keepalive interval must be 10 seconds or more.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id*
3. **keepalive** {*seconds* [*retry-count*] | **disable**}
or
no keepalive
4. **end**
or
commit
5. **show interfaces** *type interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`interface pos` *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0`
		Specifies the POS interface name and notation *rack/slot/module/port* and enters interface configuration mode.

Step 3	`keepalive {`*`seconds`* `[`*`retry-count`*`] | disable}`
	or
	`no keepalive`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# keepalive 3`
	or
	`RP/0/RSP0/CPU0:router(config-if)# no keepalive`
		Specifies the number of seconds between keepalive messages, and optionally the number of keepalive messages that can be sent to a peer without a response before transitioning the link to the down state.
		• Use the **keepalive disable** command, the **no keepalive**, or the **keepalive** command with an argument of 0 to disable the keepalive feature entirely.
		• If keepalives are configured on an interface, use the **no keepalive** command to disable the keepalive feature before you configure Frame Relay encapsulation on that interface.

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show interfaces pos** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router# show interfaces POS 0/3/0/0`	(Optional) Verifies the interface configuration.

How to Configure a Layer 2 Attachment Circuit

The Layer 2 AC configuration tasks are described in the following procedures:

- Creating a Layer 2 Frame Relay Subinterface with a PVC
- Configuring Optional Layer 2 PVC Parameters

Note After you configure an interface for Layer 2 switching, no routing commands such as **ipv4 address** are permissible.

Note Layer 2 ACs are not supported on interfaces configured with HDLC or PPP encapsulation.

Creating a Layer 2 Frame Relay Subinterface with a PVC

The procedure in this section creates a Layer 2 Frame Relay subinterface with a PVC.

Prerequisites

Before you can create a subinterface on a POS interface, you must bring up a POS interface, as described in the "Bringing Up a POS Interface" section on page 491.

Note You must skip Step 4 of the "Bringing Up a POS Interface" configuration steps when configuring an interface for Layer 2 switching. The **ipv4 address** command is not permissible on Frame Relay encapsulated interface.

Restrictions

- Only one PVC can be configured for each subinterface.
- The configuration on both ends of the PVC must match for the connection to operate properly.
- The **ipv4 address** command is not permissible on Frame Relay encapsulated interface. Any previous configuration of an IP address must be removed before you can configure an interface for Layer 2 transport mode.
- Layer 2 configuration is supported on Frame Relay PVCs only. Layer 2 Port mode, where Layer 2 configuration is applied directly under the main POS interface, is not supported.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **end**
   or
   commit
5. Repeat Step 1 through Step 4 to bring up the subinterface and any associated PVC at the other end of the AC.

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface pos`** *`interface-path-id.subinterface`* **`l2transport`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface pos 0/3/0/0.1 l2transport`
		Creates a subinterface and enters POS subinterface configuration mode for that subinterface.
		Note The *subinterface* must be unique to any other subinterfaces configured under a single main interface.

Step 3	**`pvc`** `dlci`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# pvc 100`
		Creates a Frame Relay permanent virtual circuit (PVC) and enters Layer 2 transport PVC configuration mode.
		Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007.
		Note Only one PVC is allowed per subinterface.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# end`
	or
	`RP/0/RSP0/CPU0:router(config-fr-vc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	Repeat Step 1 through Step 4 to bring up the subinterface and any associated PVC at the other end of the AC.	Brings up the AC. **Note** The configuration on both ends of the AC must match.

What to Do Next

- To configure optional subinterface parameters, see the "Configuring Optional Layer 2 Subinterface Parameters" section on page 508.

- To configure optional PVC parameters, see the "Configuring Optional Layer 2 PVC Parameters" section on page 506.
- For more information about configuring Layer 2 services on the Cisco ASR 9000 Series Router, see the "Implementing Point to Point Layer 2 Services" module of the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*.

Configuring Optional Layer 2 PVC Parameters

This task describes the commands you can use to modify the default configuration on a Frame Relay Layer 2 PVC.

Prerequisites

You must create the PVC on a Layer 2 subinterface, as described in the "Creating a Layer 2 Frame Relay Subinterface with a PVC" section on page 504.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end**
or
commit
7. Repeat Step 1 through Step 5 to configure the PVC at the other end of the AC.
8. **show policy-map interface pos** *interface-path-id.subinterface* {**input** | **output**}
or
show policy-map type qos interface pos *interface-path-id.subinterface* {**input** | **output**}

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface pos`** *`interface-path-id.subinterface`* **`l2transport`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface pos 0/6/0/1.10 l2transport`
		Enters POS subinterface configuration mode for a Layer 2 Frame Relay subinterface.

	Command or Action	Purpose

Step 3	**`pvc`** *`dlci`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# pvc 100`
		Enters Frame Relay PVC configuration mode for the specified PVC.
		Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007.

Step 4	**`encap`** `{`**`cisco`** `|` **`ietf`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# encap ietf`
		Configures the encapsulation for a Frame Relay PVC.
		The encapsulation type must match on both ends of the PVC.

Step 5	**`service-policy`** `{`**`input`** `|` **`output`**`}` *`policy-map`*
	Example:
	`RP/0/RSP0/CPU0:router (config-fr-vc)#`
	`service-policy output policy1`
		Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.
		Note For information on creating and configuring policy maps, refer to the *Cisco IOS XR Modular Quality of Service Configuration Guide*,

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-pos-l2transport-pv`
	`c)# end`
	or
	`RP/0/RSP0/CPU0:router(config-pos-l2transport-pv`
	`c)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before`
		`exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 7	Repeat Step 1 through Step 5 to configure the PVC at the other end of the AC.	Brings up the AC. **Note** The configuration on both ends of the connection must match.

Step 8	**`show policy-map interface pos`** *`interface-path-id.subinterface`* `{`**`input`** `|` **`output`**`}`
	or
	`show policy-map type qos interface pos` *`interface-path-id.subinterface`* `{`**`input`** `|` **`output`**`}`
	Example:
	`RP/0/RSP0/CPU0:router# show policy-map interface pos 0/6/0/1.10 output`
	or
	`RP/0/RSP0/CPU0:router# show policy-map type qos interface pos 0/6/0/1.10 output`
		(Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface.

Configuring Optional Layer 2 Subinterface Parameters

This task describes the commands you can use to modify the default configuration on a Frame Relay Layer 2 subinterface.

Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a Layer 2 subinterface, as described in the "Creating a Layer 2 Frame Relay Subinterface with a PVC" section on page 504.

Restrictions

In most cases, the MTU that is configured under the subinterface has priority over the MTU that is configured under the main interface. The exception to this rule is when the subinterface MTU is higher than main interface MTU. In such cases, the subinterface MTU displays the configured value in the CLI output, but the actual operational MTU is the value that is configured under the main interface value. To avoid confusion when troubleshooting and optimizing your Layer 2 connections, we recommend always configuring a higher MTU on main interface.

SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface*
3. **mtu** *value*
4. **end**
   or
   commit
5. Repeat Step 1 through Step 4 to configure the subinterface at the other end of the AC.

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface pos`** *`interface-path-id.subinterface`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface pos 0/3/0/1.1`
		Enters POS subinterface configuration mode for a Layer 2 Frame Relay subinterface.

Step 3	**`mtu`** *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# mtu 5000`
		(Optional) Configures the MTU value. Range is from 64 through 65535.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-pos-l2transport-pvc)# end`
	or
	`RP/0/RSP0/CPU0:router(config-pos-l2transport-pvc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	Repeat Step 1 through Step 4 to configure the PVC at the other end of the AC.	Brings up the AC. **Note** The configuration on both ends of the connection must match.

Configuration Examples for POS Interfaces

This section provides the following configuration examples:

- Bringing Up and Configuring a POS Interface with Cisco HDLC Encapsulation: Example, page 510
- Configuring a POS Interface with Frame Relay Encapsulation: Example, page 510
- Configuring a POS Interface with PPP Encapsulation: Example, page 512

Bringing Up and Configuring a POS Interface with Cisco HDLC Encapsulation: Example

The following example shows how to bring up a basic POS interface with Cisco HDLC encapsulation:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure the interval between keepalive messages to be 10 seconds:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# keepalive 10
RP/0/RSP0/CPU0:router(config-if)# commit
```

Configuring a POS Interface with Frame Relay Encapsulation: Example

The following example shows how to create a POS interface with Frame Relay encapsulation and a point-to-point POS subinterface with a PVC on router 1:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/0.1 point-to-point
RP/0/RSP0/CPU0:router (config-subif)#ipv4 address 10.20.3.1/24
RP/0/RSP0/CPU0:router (config-subif)# pvc 100
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# show interface POS 0/3/0/0

Wed Oct  8 04:20:30.248 PST DST
POS0/3/0/0 is up, line protocol is up
  Interface state transitions: 1
  Hardware is Packet over SONET/SDH
  Internet address is 10.20.3.1/24
  MTU 4474 bytes, BW 155520 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 32, controller loopback not set,
  LMI enq sent  116, LMI stat recvd 76, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  Last clearing of "show interface" counters 00:00:06
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     1 packets input, 13 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     1 packets output, 13 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how to create a POS interface with Frame Relay encapsulation and a point-to-point POS subinterface with a PVC on router 2, which is connected to router 1:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/1
RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/RSP0/CPU0:router(config-if)# frame-relay intf-type dce
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/1.1 point-to-point
RP/0/RSP0/CPU0:router (config-subif)#ipv4 address 10.20.3.2/24
RP/0/RSP0/CPU0:router (config-subif)# pvc 100
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# show interface POS 0/3/0/1

Wed Oct  8 04:20:38.037 PST DST
POS0/3/0/1 is up, line protocol is up
  Interface state transitions: 1
  Hardware is Packet over SONET/SDH
  Internet address is 10.20.3.2/24
  MTU 4474 bytes, BW 155520 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 32, controller loopback not set,
  LMI enq sent  0, LMI stat recvd 0, LMI upd recvd 0
  LMI enq recvd 77, LMI stat sent  77, LMI upd sent  0 , DCE LMI up
  LMI DLCI 1023  LMI type is CISCO  frame relay DCE
  Last clearing of "show interface" counters 00:00:14
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2 packets input, 26 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     2 packets output, 26 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how create a Layer 2 POS subinterface with a PVC on the main POS interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/0.1 l2transport
RP/0/RSP0/CPU0:router (config-subif)# pvc 100
RP/0/RSP0/CPU0:router(config-subif)# commit
```

Configuring a POS Interface with PPP Encapsulation: Example

The following example shows how to create and configure a POS interface with PPP encapsulation:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# show interfaces POS 0/3/0/0

POS0/3/0/0 is down, line protocol is down
  Hardware is Packet over SONET
  Internet address is 172.18.189.38/27
  MTU 4474 bytes, BW 2488320 Kbit
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation PPP, crc 32, controller loopback not set, keepalive set (
10 sec)
  LCP Closed
  Closed: IPCP
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

Additional References

The following sections provide references related to POS interface configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using the Cisco IOS XR software.	*Cisco IOS XR Getting Started Guide*

Related Topic	Document Title
Cisco IOS XR AAA services configuration information	*Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference*
Information about user groups and task IDs	*Cisco IOS XR Interface and Hardware Component Command Reference*

Standards

Standards	Title
FRF.1.2	*PVC User-to-Network Interface (UNI) Implementation Agreement - July 2000*
ANSI T1.617 Annex D	—
ITU Q.933 Annex A	—

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1294	*Multiprotocol Interconnect Over Frame Relay*
RFC 1315	*Management Information Base for Frame Relay DTEs*
RFC 1490	*Multiprotocol Interconnect Over Frame Relay*
RFC 1586	*Guidelines for Running OSPF Over Frame Relay Networks*
RFC 1604	*Definitions of Managed Objects for Frame Relay Service*
RFC 2115	*Management Information Base for Frame Relay DTEs Using SMIv2*
RFC 2390	*Inverse Address Resolution Protocol*
RFC 2427	*Multiprotocol Interconnect Over Frame Relay*
RFC 2954	*Definitions of Managed Objects for Frame Relay Service*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Serial Interfaces on the Cisco ASR 9000 Series Router

This module describes the configuration of serial interfaces on the Cisco ASR 9000 Series Router.

Before you configure a serial interface, you must configure the clear channel T3/E3 controller or channelized T1/E1controller (DS0 channel) that is associated with that interface.

Feature History for Configuring Serial Controller Interfaces

Release	Modification
Release 3.3.0	This feature was introduced on the Cisco XR 12000 Series Router. Support was added on the Cisco XR 12000 Series Router for the following hardware: • Cisco XR 12000 SIP-401 • Cisco XR 12000 SIP-501 • Cisco XR 12000 SIP-601 Support was added on the Cisco XR 12000 Series Router for the following SPAs: • Cisco 2-Port and 4-Port Channelized T3/DS0 SPA • Cisco 2-Port and 4-Port T3/E3 Serial SPA
Release 3.4.0	Support for the following features was introduced: • Subinterfaces with permanent virtual circuits (PVCs) • Frame Relay encapsulation on serial main interfaces and PVCs on the following hardware: – Cisco 8-port Channelized T1/E1 SPA – Cisco 2-Port and 4-Port Channelized T3/DS0 SPA – Cisco 2-Port and 4-Port T3/E3 Serial SPA – Cisco 1-Port Channelized OC-3 SPA – Cisco 1-Port Channelized OC-12 SPA – Cisco 1-Port Channelized OC-48 SPA – Cisco 1-Port Channelized OC-12/STM-4 ISE Line Card

Release 3.4.1	This feature was introduced on the Cisco CRS-1 Router. Support was added on the Cisco CRS-1 Router for the following hardware: • Cisco CRS-1 SIP-800 • Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA Multilink PPP was supported on serial interfaces on the Cisco XR 12000 Series Router.
Release 3.5.0	Support was added on the Cisco XR 12000 Series Router for the following SPAs: • Cisco 1-Port Channelized OC-12/DS0 SPA • Cisco 1-Port Channelized OC-48/STM-16 SPA
Release 3.7.0	Support was added on the Cisco XR 12000 Series Router for the 1-Port Channelized OC-48/DS3 Line Card.
Release 3.8.0	Support was added on the Cisco XR 12000 Series Router for quality of service (QoS) on Layer 2 subinterfaces and the following line cards: • Cisco 1-Port Channelized OC-12/DS0 Line Card • Cisco 4-Port Channelized OC-12/DS3 Line Card
Release 3.9.0	Support for serial interfaces was added on the Cisco ASR 9000 Series Router for the 2-Port Channelized OC-12c/DS0 SPA.
Release 4.0.0	Support for the following features and SPAs was added on the Cisco ASR 9000 Series Router: • Support for IPv4 multicast was added for serial interfaces. For more information about multicast configuration on an interface, see the *Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide*. • IPHC was added on the Cisco 2-Port Channelized OC-12c/DS0 SPA. • Support for the Cisco 1-Port Channelized OC-48/STM-16 SPA was introduced.
Release 4.0.0	Support for fragmentation counters using the **fragment-counter** command was added for the following SPAs: • Cisco 1-Port Channelized OC-3/STM-1 SPA • Cisco 4-Port Channelized T3/DS0 SPA • Cisco 8-Port Channelized T1/E1 SPA

Release 4.0.1	Support for the following SPAs was added: • Cisco 1-Port Channelized OC-3/STM-1 SPA • Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA
Release 4.1.0	Support for the following SPAs was added: • Cisco 4-Port Channelized T3/DS0 SPA • Cisco 8-Port Channelized T1/E1 SPA Support for IPHC was added on the following SPAs: • Cisco 1-Port Channelized OC-3/STM-1 SPA • Cisco 4-Port Channelized T3/DS0 SPA • Cisco 8-Port Channelized T1/E1 SPA • Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA

Contents

Prerequisites for Configuring Serial Interfaces

Before configuring serial interfaces, ensure that the following tasks and conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You have installed a 2-Port or 4-Port Clear Channel T3/E3 SPA.
- You should have the following SIP and any one of the following SPAs installed on the Cisco ASR 9000 Series Router:
  - Cisco SIP 700 SPA Interface Processor
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 2-Port Channelized OC-12c/DS0 SPA
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
  - Cisco 2-Port or 4-Port Clear Channel T3/E3 SPA
  - Cisco 4-Port Channelized T3/DS0 SPA
  - Cisco 8-Port Channelized T1/E1 SPA
- Your hardware must support T3/E3 or T1/E1 controllers and serial interfaces.

  The following hardware supports T3/E3 controllers and serial interfaces:
  - Cisco 2-Port and 4-Port Clear Channel T3/E3 SPAs

  - Cisco 2-Port and 4-Port Channelized T3 SPAs
  - Cisco 4-Port Channelized OC-12/DS3 line cards
  - Cisco 1-Port Channelized OC-48/STM-16 SPA and line cards

  The following hardware supports T1/E1 controllers and DS0 channels:

  - Cisco 2-Port and 4-Port Channelized T3 SPAs
  - Cisco 4-Port Channelized OC-12/DS3 line cards
  - Cisco 1-Port Channelized OC-12/DS0 SPAs and line cards
  - Cisco 1-Port Channelized OC-48/DS3 SPAs and line cards
  - Cisco 1-Port Channelized OC3/STM-1 SPA
  - Cisco 8-Port Channelized T1/E1 SPA

  The following hardware supports serial interfaces:

  - Cisco 2-Port and 4-Port Clear Channel T3/E3 SPAs
  - Cisco 2-Port and 4-Port Channelized T3 SPAs
  - Cisco 4-Port Channelized OC-12/DS3 line cards
  - Cisco 1-Port Channelized OC-12/DS0 SPAs and line cards
  - Cisco 1-Port Channelized OC-48/DS3 SPAs and line cards
  - Cisco 1-Port Channelized OC3/STM-1 SPA
  - Cisco 8-Port Channelized T1/E1 SPA

Note The Cisco 2-Port and 4-Port Channelized T3 SPAs can run in clear channel mode, or they can be channelized into 28 T1 or 21 E1 controllers.

Note The Cisco 4-Port Channelized T3/DS0 SPA can run in clear channel mode, or it can be channelized into 28 T1 or 21 E1 controllers.

- You should have configured the clear channel T3/E3 controller or channelized T3 to T1/E1 controller that is associated with the serial interface you want to configure, as described in the *"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"* module in this manual.

Note On channelized T3 to T1/E1 controllers, serial interfaces are automatically created when users configure individual DS0 channel groups on the T1/E1 controllers.

Information About Configuring Serial Interfaces

To configure serial interfaces, study the following concepts:

- High-Level Overview: Serial Interface Configuration on Clear-Channel SPAs, page 503
- High-Level Overview: Serial Interface Configuration on Channelized SPAs, page 504
- Cisco HDLC Encapsulation, page 506

- PPP Encapsulation, page 506
- Keepalive Timer, page 508
- Frame Relay Encapsulation, page 509
- Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN on Frame Relay, page 510
- Default Settings for Serial Interface Configurations, page 511
- Serial Interface Naming Notation, page 511
- IPHC Overview, page 512

On the Cisco ASR 9000 Series Router, a single serial interface carries data over a single interface using PPP, Cisco HDLC, or Frame Relay encapsulation.

High-Level Overview: Serial Interface Configuration on Clear-Channel SPAs

Table 14 provides a high-level overview of the tasks required to configure a T3 serial interface on the Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA.

Table 14 Overview: Configuring a T3 Serial Interface on a Clear Channel SPA

Step	Task	Module	Section
1.	Use the **hw-module subslot** command to set serial mode for the SPA to be T3, if necessary. **Note** By default, the 2-Port and 4-Port Clear Channel T3/E3 SPA is set to run in T3 mode.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Setting the Card Type
2.	Configure the T3 controller.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Setting the Card Type
3.	Configure the serial interface that is associated with the T3 controller you configured in Step 2.	"Configuring Serial Interfaces on the Cisco ASR 9000 Series Router"	"How to Configure Serial Interfaces"

Table 15 provides a high-level overview of the tasks required to configure an E3 serial interface on a 2-Port and 4-Port Clear Channel T3/E3 SPA.

Table 15 Overview: Configuring an E3 Serial Interface on a Clear Channel SPA

Step	Task	Module	Section
1.	Use the **hw-module subslot** command to set serial mode for the SPA to be E3.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Setting the Card Type
2.	Configure the E3 controller.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Setting the Card Type
3.	Configure the serial interface that is associated with the E3 controller you configured in Step 2.	Configuring Serial Interfaces on the Cisco ASR 9000 Series Router	*How to Configure Serial Interfaces*

High-Level Overview: Serial Interface Configuration on Channelized SPAs

Table 16Table 17 provides a high-level overview of the tasks required to configure a T1 serial interface on the following SPAs and line cards.

- Cisco 2-Port and 4-Port Channelized T3 SPA
- Cisco 4-Port Channelized OC-12/DS3 line cards
- Cisco 1-Port Channelized OC-12/DS0 SPAs and line cards
- Cisco 1-Port Channelized OC-48/STM-16 SPA and line cards
- Cisco 2-Port Channelized OC-12c/DS0 SPA

Table 16 Overview: Configuring a Serial Interface on a T1 DS0 Channel

Step	Task	Module	Section
1.	Configure the T3 controller parameters and set the SPA mode to be T3. 28 T1 controllers are automatically created.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Setting the Card Type Configuring a Channelized T3 Controller
2.	Create and configure DS0 channel groups on the T1 controllers.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring a T1 Controller
3.	Configure the Serial interfaces that are associated channel groups you created in Step 2.	"Configuring Serial Interfaces on the Cisco ASR 9000 Series Router"	How to Configure Serial Interfaces

Table 17 Overview: Configuring a Serial Interface on a T1 DS0 Channel

Step	Task	Module	Section
1.	Configure the SONET controller parameters and STS stream for T3 mode.	"Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router"	Configuring SONET T3 and VT1.5-Mapped T1 Channels
2.	Configure the T3 controller parameters and set the mode to T1. 28 T1 controllers are automatically created.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring a Channelized T3 Controller
3.	Create and configure DS0 channel groups on the T1 controllers.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring a T1 Controller
4.	Configure the Serial interfaces that are associated channel groups you created in Step 2.	"Configuring Serial Interfaces on the Cisco ASR 9000 Series Router"	How to Configure Serial Interfaces

Table 18 provides a high-level overview of the tasks required to configure an E1 serial interface on the following SPAs and line cards.

- 2-Port and 4-Port Channelized T3 SPA
- 4-Port Channelized OC-12/DS3 line cards
- 1-Port Channelized OC-12/DS0 SPAs and line cards
- 1-Port Channelized OC-48/DS3 SPAs and line cards
- 1-Port Channelized OC-3/STM-1 SPA
- 2-Port Channelized OC-12c/DS0 SPA

Table 18 Overview: Configuring a Serial Interface on an E1 DS0 Channel

Step	Task	Module	Section
1.	Configure the T3 controller parameters and set the SPA mode to be E3. 21 E1 controllers are automatically created.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring a Channelized T3 Controller
2.	Create and configure DS0 channel groups on the E1 controllers.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring an E1 Controller
3.	Configure the Serial interfaces that are associated channel groups you created in Step 2.	Configuring Serial Interfaces on the Cisco ASR 9000 Series Router	How to Configure Serial Interfaces

Table 19 Overview: Configuring a Serial Interface on a E1 DS0 Channel

Step	Task	Module	Section
1.	Configure the SONET controller parameters and STS stream for T3 mode.	"Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router"	Configuring SONET T3 and VT1.5-Mapped T1 Channels
2.	Configure the T3 controller parameters and set the mode to E1. 21 E1 controllers are automatically created.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring a Channelized T3 Controller
3.	Create and configure DS0 channel groups on the E1 controllers.	"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"	Configuring an E1 Controller
4.	Configure the Serial interfaces that are associated channel groups you created in Step 2.	"Configuring Serial Interfaces on the Cisco ASR 9000 Series Router"	How to Configure Serial Interfaces

Table 20 provides a high-level overview of the tasks required to configure a T3 serial interface on the 1-Port Channelized OC-48/STM-16 SPA

Table 20 Overview: Configuring a Serial Interface on a T3 Channel

Step	Task	Module	Section
1.	Configure the SONET controller parameters and STS stream.	"Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router"	Configuring a Clear Channel SONET Controller for T3
2.	Configure the STS stream mode for T3 and configure the T3 controller parameters.	"Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router"	Configuring a Clear Channel SONET Controller for T3
3.	Configure the Serial interfaces.	"Configuring Serial Interfaces on the Cisco ASR 9000 Series Router"	How to Configure Serial Interfaces

Cisco HDLC Encapsulation

Cisco High-Level Data Link Controller (HDLC) is the Cisco proprietary protocol for sending data over synchronous serial links using HDLC. Cisco HDLC also provides a simple control protocol called Serial Line Address Resolution Protocol (SLARP) to maintain serial link keepalives. HDLC is the default encapsulation type for serial interfaces under Cisco IOS XR software. Cisco HDLC is the default for data encapsulation at Layer 2 (data link) of the Open System Interconnection (OSI) stack for efficient packet delineation and error control.

Note Cisco HDLC is the default encapsulation type for the serial interfaces.

Cisco HDLC uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 508.

Note Use the **debug chdlc slarp packet** command to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

PPP Encapsulation

PPP is a standard protocol used to send data over synchronous serial links. PPP also provides a Link Control Protocol (LCP) for negotiating properties of the link. LCP uses echo requests and responses to monitor the continuing availability of the link.

Note When an interface is configured with PPP encapsulation, a link is declared down, and full LCP negotiation is re-initiated after five ECHOREQ packets are sent without receiving an ECHOREP response.

PPP provides the following Network Control Protocols (NCPs) for negotiating properties of data protocols that will run on the link:

- IP Control Protocol (IPCP) to negotiate IP properties
- Multiprotocol Label Switching control processor (MPLSCP) to negotiate MPLS properties
- Cisco Discovery Protocol control processor (CDPCP) to negotiate CDP properties
- IPv6CP to negotiate IP Version 6 (IPv6) properties
- Open Systems Interconnection control processor (OSICP) to negotiate OSI properties

PPP uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 508.

PPP supports the following authentication protocols, which require a remote device to prove its identity before allowing data traffic to flow over a connection:

- Challenge Handshake Authentication Protocol (CHAP)—CHAP authentication sends a challenge message to the remote device. The remote device encrypts the challenge value with a shared secret and returns the encrypted value and its name to the local router in a response message. The local router attempts to match the name of the remote device with an associated secret stored in the local username or remote security server database; it uses the stored secret to encrypt the original challenge and verify that the encrypted values match.
- Microsoft Challenge Handshake Authentication Protocol (MS-CHAP)—MS-CHAP is the Microsoft version of CHAP. Like the standard version of CHAP, MS-CHAP is used for PPP authentication; in this case, authentication occurs between a personal computer using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server.
- Password Authentication Protocol (PAP)—PAP authentication requires the remote device to send a name and a password, which are checked against a matching entry in the local username database or in the remote security server database.

Note For more information on enabling and configuring PPP authentication protocols, see the *"Configuring PPP on the Cisco ASR 9000 Series Router"* module in this manual.

Use the **ppp authentication** command in interface configuration mode to enable CHAP, MS-CHAP, and PAP on a serial interface.

Note Enabling or disabling PPP authentication does not effect the local router's willingness to authenticate itself to the remote device.

Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) is supported on the following SPAs:

- 1-Port Channelized OC-12/DS0 SPAs and line cards
- 2-Port and 4-Port Channelized T3 SPAs
- 8-Port Channelized T1/E1 SPA
- 1-Port Channelized OC-3/STM-1 SPA
- 2-Port Channelized OC-12/DS0 SPA

MLPPP provides a method for combining multiple physical links into one logical link. The implementation of MLPPP combines multiple PPP serial interfaces into one multilink interface. MLPPP performs the fragmenting, reassembling, and sequencing of datagrams across multiple PPP links.

MLPPP provides the same features that are supported on PPP Serial interfaces with the exception of QoS. It also provides the following additional features:

- Fragment sizes of 128, 256, and 512 bytes
- Long sequence numbers (24-bit)
- Lost fragment detection timeout period of 80 ms
- Minimum-active-links configuration option
- LCP echo request/reply support over multilink interface
- Full T1 and E1 framed and unframed links

For more information about configuring MLPPP on a serial interface, see the *"Configuring PPP on the Cisco ASR 9000 Series Router"* module in this document.

Keepalive Timer

Cisco keepalives are useful for monitoring the link state. Periodic keepalives are sent to and received from the peer at a frequency determined by the value of the keepalive timer. If an acceptable keepalive response is not received from the peer, the link makes the transition to the down state. As soon as an acceptable keepalive response is obtained from the peer or if keepalives are disabled, the link makes the transition to the up state.

Note The **keepalive** command applies to serial interfaces using HDLC or PPP encapsulation. It does not apply to serial interfaces using Frame Relay encapsulation.

For each encapsulation type, a certain number of keepalives ignored by a peer triggers the serial interface to transition to the down state. For HDLC encapsulation, three ignored keepalives causes the interface to be brought down. For PPP encapsulation, five ignored keepalives causes the interface to be brought down. ECHOREQ packets are sent out only when LCP negotiation is complete (for example, when LCP is open).

Use the **keepalive** command in interface configuration mode to set the frequency at which LCP sends ECHOREQ packets to its peer. To restore the system to the default keepalive interval of 10 seconds, use the **keepalive** command with **no** argument. To disable keepalives, use the **keepalive disable** command. For both PPP and Cisco HDLC, a keepalive of 0 disables keepalives and is reported in the **show running-config** command output as **keepalive disable**.

Note During Minimal Disruptive Restart (MDR) keepalives may fail. Therefore the keepalive timers, on both ends, should be either disabled or set to an interval longer than the MDR time.

Note Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend disabling keepalives on a Cisco XR 12000 Series Router.

Note Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend configuring a keepalive interval of 10 seconds or more on a Cisco CRS-1 Router.

When LCP is running on the peer and receives an ECHOREQ packet, it responds with an echo reply (ECHOREP) packet, regardless of whether keepalives are enabled on the peer.

Keepalives are independent between the two peers. One peer end can have keepalives enabled; the other end can have them disabled. Even if keepalives are disabled locally, LCP still responds with ECHOREP packets to the ECHOREQ packets it receives. Similarly, LCP also works if the period of keepalives at each end is different.

Note Use the **debug chdlc slarp packet** command and other Cisco HDLC **debug** commands to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

Frame Relay Encapsulation

When Frame Relay encapsulation is enabled on a serial interface, the interface configuration is hierarchical and comprises the following elements:

1. The serial main interface comprises the physical interface and port. If you are not using the serial interface to support Cisco HDLC and PPP encapsulated connections, then you must configure subinterfaces with permanent virtual circuits (PVCs) under the serial main interface. Frame Relay connections are supported on PVCs only.
2. Serial subinterfaces are configured under the serial main interface. A serial subinterface does not actively carry traffic until you configure a PVC under the serial subinterface. Layer 3 configuration typically takes place on the subinterface.
3. Point-to-point PVCs are configured under a serial subinterface. You cannot configure a PVC directly under a main interface. A single point-to-point PVC is allowed per subinterface. PVCs use a predefined circuit path and fail if the path is interrupted. PVCs remain active until the circuit is removed from either configuration. Connections on the serial PVC support Frame Relay encapsulation only.

Note The administrative state of a parent interface drives the state of the subinterface and its PVC. When the administrative state of a parent interface or subinterface changes, so does the administrative state of any child PVC configured under that parent interface or subinterface.

To configure Frame Relay encapsulation on serial interfaces, use the **encapsulation frame-relay** command.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (default)
- IETF

Use the **encap** command in PVC configuration mode to configure Cisco or IETF encapsulation on a PVC. If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.

Note Cisco encapsulation is required on serial main interfaces that are configured for MPLS. IETF encapsulation is not supported for MPLS.

Before you configure Frame Relay encapsulation on an interface, you must verify that all prior Layer 3 configuration is removed from that interface. For example, you must ensure that there is no IP address configured directly under the main interface; otherwise, any Frame Relay configuration done under the main interface will not be viable.

LMI on Frame Relay Interfaces

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay UNI interface. By default, **cisco** LMI is enabled on all PVCs. However, you can modify the default LMI type to be ANSI or Q.933, as described in the *"Modifying the Default Frame Relay Configuration on an Interface"* section of the *"Configuring Frame Relay on the Cisco ASR 9000 Series Router"* module in this manual.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA:

```
(MTU - 13)/8 = maximum number of PVCs
```

Note The default setting of the **mtu** command for a serial interface is 1504 bytes. Therefore, the default numbers of PVCs supported on a serial interface configured with **cisco** LMI is 186.

Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN on Frame Relay

The Layer 2 Tunnel Protocol Version 3 (L2TPv3) feature defines the L2TP protocol for tunneling Layer 2 payloads over an IP core network using Layer 2 virtual private networks (VPNs).

L2TPv3 is a tunneling protocol used for transporting Layer 2 protocols. It can operate in a number of different configurations and tunnel a number of different Layer 2 protocols and connections over a packet-switched network.

Before you can configure L2TPv3, you need to configure a connection between the two attachment circuits (ACs) that will host the L2TPv3 psuedowire. Cisco IOS XR software supports a point-to-point, end-to-end service, where two ACs are connected together.

This module describes how to configure a Layer 2 AC on a Frame Relay encapsulated serial interface.

Note Serial interfaces support DLCI mode layer 2 ACs only; layer 2 port mode ACs are not supported on serial interfaces.

For detailed information about configuring L2TPv3 in your network, see the "*Implementing Layer 2 Tunnel Protocol Version 3*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router*. For detailed information about configuring L2VPNs, see the "*Implementing MPLS Layer 2 VPNs*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router*.

For detailed information about configuring L2TPv3 in your network, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router*. For detailed information about configuring L2VPNs, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router*.

Default Settings for Serial Interface Configurations

When an interface is enabled on a T3/E3 SPA, and no additional configuration commands are applied, the default interface settings shown in Table 21 are present. These default settings can be changed by configuration.

Table 21 Serial Interface Default Settings

Parameter	Configuration File Entry	Default Settings
Keepalive **Note** The **keepalive** command applies to serial interfaces using HDLC or PPP encapsulation. It does not apply to serial interfaces using Frame Relay encapsulation.	**keepalive** [**disable**] **no keepalive**	keepalive 10 seconds
Encapsulation	**encapsulation** [**hdlc** | **ppp** | **frame-relay** [**ietf**]]	hdlc
Maximum transmission unit (MTU)	**mtu** *bytes*	1504 bytes
Cyclic redundancy check (CRC)	**crc** [**16** | **32]**	16
Data stream inversion on a serial interface	**invert**	Data stream is not inverted
Payload scrambling (encryption)	**scramble**	Scrambling is disabled.
Number of High-Level Data Link Control (HDLC) flag sequences to be inserted between the packets	**transmit-delay**	Default is 0 (disabled).

Note Default settings do not appear in the output of the **show running-config** command.

Serial Interface Naming Notation

The naming notation for serial interfaces on a clear channel SPA is *rack/slot/module/port*, as shown in the following example:

```
interface serial 0/0/1/2
```

The naming notation for T1, E1, and DS0 interfaces on a channelized SPA is *rack/slot/module/port/channel-num:channel-group-number*, as shown in the following example:

```
interface serial 0/0/1/2/4:3
```

If a subinterface and PVC are configured under the serial interface, then the router includes the subinterface number at the end of the serial interface address. In this case, the naming notation is *rack/slot/module/port*[*/channel-num:channel-group-number*].*subinterface,* as shown in the following examples:

```
interface serial 0/0/1/2.1
interface serial 0/0/1/2/4:3.1
```

Note A slash between values is required as part of the notation.

The naming notation syntax for serial interfaces is as follows:

- *rack*: Chassis number of the rack.
- *slot*: Physical slot number of the modular services card or line card.
- *module*: Module number. Shared port adapters (SPAs) are referenced by their subslot number.
- *port*: Physical port number of the controller.
- *channel-num*: T1 or E1 channel number. T1 channels range from 0 to 23; E1 channels range from 0 to 31.
- *channel-group-number*: Time slot number. T1 time slots range from 1 to 24; E1 time slots range from 1 to 31. The *channel-group-number* is preceded by a colon and not a slash.
- *subinterface*: Subinterface number.

Use the question mark (**?**) online help function following the **serial** keyword to view a list of all valid interface choices.

IPHC Overview

IP header compression (IPHC) is based on the premise that most of the headers in the packets of a particular transmission remain constant throughout the flow. Only a few fields in the headers of related packets change during a flow.

IPHC compresses these headers so that the compressed header contains only the fields that change from packet to packet. All fields that remain the same from packet to packet are eliminated in the compressed headers. Full headers are sent between compressed headers.

Full headers are uncompressed headers that contain all the original header fields along with additional information (context ID) to identify the flow. The interval at which full headers are sent between compressed packets is configurable using the **refresh max-period** and **refresh max-time** commands.

IPHC contexts are used by the compressor (sender) and decompressor (receiver) of compressed packets to encode and decode the packets in a flow. A context is stored on the compressor and decompressor and is used in the delta calculation at both ends. The number of contexts allowed on a particular interface is configurable. The maximum size of the header that can be compressed is also configurable.

IPHC supports the compression and decompression of RTP and UDP traffic and the decompression of CN on TCP and CTCP traffic.

Users may choose one of the following types of compression formats:

- Internet Engineering Task Force (IETF) standard format.
  Uses RFC2507 and RFC2508 compression schemes.
- IPHC format.
  Provides options similar to IETF.

Table 22 shows the IPHC features, the values of the features, and their defaults:

Table 22 IPHC features and default setttings

IPHC Feature	Values	Defaults
TCP contexts	0 to 255	1
Non-TCP contexts	1 to 6000	16
Compression Format Options	IETF or IPHC	—
Feedback Messages	Enable or Disable	Enabled
Maximum Refresh Period Size	1 to 65535 packets	256
Maximum Refresh Time Period	0 to 255 seconds	5
Maximum Header Size	20 to 40 bytes	40
Real Time Protocol (RTP)	Enable or Disable	Enabled
Refresh RTP	Enable or Disable	Disable

Currently, only IPv4 unicast packets with UDP in the protocol field of the IP header are compressed.

IPHC is configured on an interface as follows:

- Configure the IPHC slot level command
- Create an IPHC profile
- Configure IPHC attributes in the profile
- Attach the profile to an interface

IPHC profiles must contain the **rtp** command to enable Real Time Protocol (RTP) on the interface, or the profile is not enabled. The **refresh rtp** command must be used to enable the configured refresh settings for RTP packets. By default, refresh RTP is disabled and only the first packet in the flow is sent as a ‘full-header’ packet.

If some attributes, such as feedback messages, maximum refresh period size, maximum refresh time period, and maximum header size, are not configured in the profile, the default values for those attributes apply when the profile is enabled on the interface.

Currently, IPHC is supported only on serial interfaces with PPP encapsulation and on multilink with PPP encapsulation interfaces.

IPHC is typically configured between the Customer Edge (CE) and Provide Edge (PE) ends of an interface and must be configured at both ends of the interface to work. The PPP protocol negotiates the IPHC specific parameters between the two ends of the interface and settles on the lowest value configured between the two ends.

QoS and IPHC

An IPHC profile can be enabled on an interface so that the IPHC profile applies only to packets that match a Quality of Service (QoS) service policy. In this case, the QoS service-policy class attributes determine which packets are compressed. This allows users to fine tune IPHC with greater granularity.

Policy maps are attached to an interface using the **service-policy** command. IPHC action applies only to output service policies. IPHC is not supported on input service policies.

The user can configure IPHC using QoS as follows:

- Create a QoS **policy-map** with the **compress header ip** action.
- Attach the IPHC profile to the interface using the **ipv4 iphc profile** *profile_name* **mode service-policy** command.
- Attach the QoS **policy-map** with **compress header ip** action using the **service-policy output** command.

See "IPHC on a Serial Interface with MLPPP/LFI and QoS Configuration: Example" section on page 547 for an example of how to configure IPHC using QoS.

For complete information on configuring QoS, refer to the *Cisco XR 12000 Series Router Modular Quality of Service Configuration Guide* and the *Cisco XR 12000 Series Router Modular Quality of Service Command ReferenceCisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide* and the *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Command Reference*.

How to Configure Serial Interfaces

After you have configured a channelized or clear channel T3/E3 controller, as described in the "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router" module in this document, you can configure the serial interfaces associated with that controller.

The following tasks describe how to configure a serial interface:

- Bringing Up a Serial Interface, page 514
- Configuring Optional Serial Interface Parameters, page 518
- Creating a Point-to-Point Serial Subinterface with a PVC, page 521
- Configuring Optional PVC Parameters, page 524
- Modifying the Keepalive Interval on Serial Interfaces, page 526
- How to Configure a Layer 2 Attachment Circuit, page 528
  - Creating a Serial Layer 2 Subinterface with a PVC, page 529
  - Configuring Optional Serial Layer 2 PVC Parameters, page 531
- Configuring IPHC, page 533
  - Configuring the IPHC Slot Level Command, page 534
  - Configuring an IPHC Profile, page 536
  - Configuring an IPHC Profile, page 538
  - Enabling an IPHC Profile on an Interface, page 541

Bringing Up a Serial Interface

This task describes the commands used to bring up a serial interface.

Prerequisites

The Cisco XR 12000 Series Router must have at least one of the SIPs and one of the SPAs or line cards installed and be running Cisco IOS XR software:

- Cisco XR 12000 SIP-401
- Cisco XR 12000 SIP-501
- Cisco XR 12000 SIP-601
- 2-Port and 4-Port T3/E3 Serial SPA
- 2-Port and 4-Port Channelized T3/DS0 Serial SPA
- 4-Port Channelized OC-12/DS3 line cards
- 1-Port Channelized OC-12/DS0 SPAs and line cards
- 1-Port Channelized OC-48/DS3 SPAs and line cards
- 1-Port Channelized OC3/STM-1 SPA
- 8-Port Channelized T1/E1 SPA

The Cisco CRS-1 Router must have the following SIP and SPA installed and running Cisco IOS XR software:

- Cisco CRS-1 SIP-800
- 2-Port and 4-Port T3/E3 Serial SPA

The Cisco ASR 9000 Series Router must have the following SIP and at least one of the following SPAs installed and running Cisco IOS XR software:

- SIP 700 SPA Interface Processor
- 1-Port Channelized OC-3/STM-1 SPA
- 2-Port Channelized OC-12c/DS0 SPA
- 1-Port Channelized OC-48/STM-16 SPA
- 4-Port Channelized T3/DS0 SPA
- 2-Port and 4-Port Clear Channel T3/E3 SPA
- 8-Port Channelized T1/E1 SPA

Restrictions

The configuration on both ends of the serial connection must match for the interface to be active.

SUMMARY STEPS

1. **show interfaces**
2. **configure**
3. **interface serial** *interface-path-id*
4. **ipv4 address** *ip-address*
5. **no shutdown**
6. **end**
   or
   commit

7. **exit**
8. **exit**
9. Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
10. **show ipv4 interface brief**
11. **show interfaces serial** *interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show interfaces** **Example:** `RP/0/0RP0RSP0/CPU0:router# show interfaces`	(Optional) Displays configured interfaces. • Use this command to also confirm that the router recognizes the PLIM card.
Step 2	**configure** **Example:** `RP/0/0RP0RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**interface serial** *interface-path-id* **Example:** `RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/0`	Specifies the serial interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 4	**ipv4 address** *ip-address* **Example:** `RP/0/0RP0RSP0/CPU0:router(config-if)# ipv4 address 10.1.2.1 255.255.255.224`	Assigns an IP address and subnet mask to the interface. **Note** Skip this step if you are configuring Frame Relay encapsulation on this interface. For Frame Relay, the IP address and subnet mask are configured under the subinterface.
Step 5	**no shutdown** **Example:** `RP/0/0RP0RSP0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming the parent SONET layer is not configured administratively down).

	Command or Action	Purpose

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-if)# end`
	or
	`RP/0/0RP0RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 7	**`exit`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-if)# exit`
		Exits interface configuration mode and enters global configuration mode.

Step 8	**`exit`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config)# exit`
		Exits global configuration mode and enters EXEC mode.

Step 9	**`show interfaces`**
	`configure`
	`interface serial` *`interface-path-id`*
	`no shut`
	`exit`
	`exit`
	Example:
	`RP/0/0RP0RSP0/CPU0:router# show interfaces`
	`RP/0/0RP0RSP0/CPU0:router# configure`
	`RP/0/0RP0RSP0/CPU0:router (config)# interface serial 0/1/0/1`
	`RP/0/0RP0RSP0/CPU0:router(config-if)# ipv4 address 10.1.2.2 255.255.255.224`
	`RP/0/0RP0RSP0/CPU0:router (config-if)# no shutdown`
	`RP/0/0RP0RSP0/CPU0:router (config-if)# commit`
	`RP/0/0RP0RSP0/CPU0:router (config-if)# exit`
	`RP/0/0RP0RSP0/CPU0:router (config)# exit`
		Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
		Note The configuration on both ends of the serial connection must match.

	Command or Action	Purpose
Step 10	**show ipv4 interface brief** **Example:** `RP/0/0RP0RSP0/CPU0:router # show ipv4 interface brief`	Verifies that the interface is active and properly configured. If you have brought up a serial interface properly, the "Status" field for that interface in the **show ipv4 interface brief** command output displays "Up."
Step 11	**show interfaces serial** *interface-path-id* **Example:** `RP/0/0RP0RSP0/CPU0:router# show interfaces serial 0/1/0/0`	(Optional) Displays the interface configuration.

What to Do Next

To modify the default configuration of the serial interface you just brought up, see the "Configuring Optional Serial Interface Parameters" section on page 518.

Configuring Optional Serial Interface Parameters

This task describes the commands used to modify the default configuration on a serial interface.

Prerequisites

Before you modify the default serial interface configuration, you must bring up the serial interface and remove the shutdown configuration, as described in the "Bringing Up a Serial Interface" section on page 514.

Restrictions

The configuration on both ends of the serial connection must match for the interface to be active.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id*
3. **encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]
4. **serial**
5. **crc** *length*
6. **invert**
7. **scramble**
8. **transmit-delay** *hdlc-flags*
9. **end**
   or
   commit
10. **exit**

11. **exit**
12. **exit**
13. **show interfaces serial** [*interface-path-id*]

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface serial`** *`interface-path-id`*
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/0`
		Specifies the serial interface name and notation *rack/slot/module/port,* and enters interface configuration mode.

Step 3	**`encapsulation`** `[`**`hdlc`** `|` **`ppp`** `|` **`frame-relay`** `[`**`IETF`**`]`
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config-if)# encapsulation hdlc`
		(Optional) Configures the interface encapsulation parameters and details such as HDLC, PPP or Frame Relay.
		Note The default encapsulation is **hdlc**.

Step 4	**`serial`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config-if)# serial`
		(Optional) Enters serial submode to configure the serial parameters.

Step 5	**`crc`** *`length`*
	Example:
	`RP/0/0RP0RSP0/CPU0:ios(config-if-serial)# crc 32`
		(Optional) Specifies the length of the cyclic redundancy check (CRC) for the interface. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode.
		Note The default is CRC length is 16.

Step 6	**`invert`**
	Example:
	`RP/0/0RP0RSP0/CPU0:ios(config-if-serial)# inverts`
		(Optional) Inverts the data stream.

Step 7	**`scramble`**
	Example:
	`RP/0/0RP0RSP0/CPU0:ios(config-if-serial)# scramble`
		(Optional) Enables payload scrambling on the interface.
		Note Payload scrambling is disabled on the interface.

Step 8	**`transmit-delay`** *`hdlc-flags`*
	Example:
	`RP/0/0RP0RSP0/CPU0:ios(config-if-serial)# transmit-delay 10`
		(Optional) Specifies a transmit delay on the interface. Values can be from 0 to 128.
		Note Transmit delay is disabled by default (the transmit delay is set to **0**).

	Command or Action	Purpose

Step 9	**`end`**
	or
	`commit`
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-if)# end`
	or
	`RP/0/0RP0RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 10	**`exit`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config-if-serial)# exit`
		Exits serial configuration mode.

Step 11	**`exit`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-if)# exit`
		Exits interface configuration mode and enters global configuration mode.

Step 12	**`exit`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config)# exit`
		Exits global configuration mode and enters EXEC mode.

Step 13	**`show interfaces serial`** [*`interface-path-id`*] **Example:** `RP/0/0RP0RSP0/CPU0:router# show interface serial 0/1/0/0`	(Optional) Displays general information for the specified serial interface.

What to Do Next

- To create a point-to-point Frame Relay subinterface with a PVC on the serial interface you just brought up, see the "Creating a Point-to-Point Serial Subinterface with a PVC" section on page 521.
- To configure PPP authentication on serial interfaces with PPP encapsulation, see the *"Configuring PPP on the Cisco ASR 9000 Series Router"* module later in this manual.

- To modify the default keepalive configuration, see the "Modifying the Keepalive Interval on Serial Interfaces" section on page 526.
- To modify the default Frame Relay configuration on serial interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" section of the *"Configuring Frame Relay on the Cisco ASR 9000 Series Router"* module.

Creating a Point-to-Point Serial Subinterface with a PVC

The procedure in this section creates a point-to-point serial subinterface and configures a permanent virtual circuit (PVC) on that serial subinterface.

Note Subinterface and PVC creation is supported on interfaces with Frame Relay encapsulation only.

Prerequisites

Before you can create a subinterface on a serial interface, you must bring up the main serial interface with Frame Relay encapsulation, as described in the "Bringing Up a Serial Interface" section on page 514.

Restrictions

Only one PVC can be configured for each point-to-point serial subinterface.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *dlci*
5. **end**
   or
   commit
6. Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection.

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface serial`** *`interface-path-id.subinterface`* **`point-to-point`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config)# interface serial 0/1/0/0.1`
		Enters serial subinterface configuration mode.

Step 3	**`ipv4 address`** *`ipv4_address/prefix`*
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24`
		Assigns an IP address and subnet mask to the subinterface.

Step 4	**`pvc`** *`dlci`*
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-subif)# pvc 20`
		Creates a serial permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.
		Replace *dlci* with a PVC identifier, in the range from 16 to 1007.
		Note Only one PVC is allowed per subinterface.

	Command or Action	Purpose

Step 5	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-subif)# end`
	or
	`RP/0/0RP0RSP0/CPU0:router(config-subif)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before`
		`exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**`configure`**
	`interface serial` *`interface-path-id`*
	`pvc` *`dlci`*
	`commit`
	
	Example:
	`RP/0/0RP0RSP0/CPU0:router# configure`
	`RP/0/0RP0RSP0/CPU0:router (config)# interface`
	`serial 0/1/0/1.1`
	`RP/0/0RP0RSP0/CPU0:router (config-subif)#ipv4`
	`address 10.46.8.5/24`
	`RP/0/0RP0RSP0/CPU0:router (config-subif)# pvc`
	`20`
	`RP/0/0RP0RSP0/CPU0:router (config-fr-vc)#`
	`commit`
		Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection.
		Note The DLCI (or PVC identifier) must match on both ends of the subinterface connection.
		Note When assigning an IP address and subnet mask to the subinterface at the other end of the connection, keep in mind that the addresses at both ends of the connection must be in the same subnet.

What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional Serial Interface Parameters" section on page 518.
- To modify the default Frame Relay configuration on serial interfaces that have Frame Relay encapsulation enabled, see the"Modifying the Default Frame Relay Configuration on an Interface" section of the *"Configuring Frame Relay on the Cisco ASR 9000 Series Router"* module.
- To attach a Layer 3 QOS service policy to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.

Configuring Optional PVC Parameters

This task describes the commands you can use to modify the default configuration on a serial PVC.

For additional information about Frame Relay options, see the "Configuring Frame Relay on Cisco IOS XR Software" module in the *Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router.*

For additional information about Frame Relay options, see the "Configuring Frame Relay on the Cisco ASR 9000 Series Router" module in the *Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco ASR 9000 Series Router.*

Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a serial subinterface, as described in the "Creating a Point-to-Point Serial Subinterface with a PVC" section on page 521.

Restrictions

- The DLCI (or PVI identifier) must match on both ends of the PVC for the connection to be active.
- To change the PVC DLCI, you must delete the PVC and then add it back with the new DLCI.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface*
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end**
   or
   commit
7. Repeat Step 1 through Step 6 to configure the PVC at the other end of the connection.
8. **show frame-relay pvc** *dlci-number*
9. **show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**}
   or
   show policy-map type qos interface serial *interface-path-id.subinterface* {**input** | **output**}

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0RP0RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *interface-path-id.subinterface* **Example:** `RP/0/0RP0RSP0/CPU0:router (config)# interface serial 0/1/0/0.1`	Enters serial subinterface configuration mode.
Step 3	**pvc** *dlci* **Example:** `RP/0/0RP0RSP0/CPU0:router (config-subif)# pvc 20`	Enters subinterface configuration mode for the PVC.
Step 4	**encap** [**cisco** | **ietf**] **Example:** `RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# encap ietf`	(Optional) Configures the encapsulation for a Frame Relay PVC. **Note** If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.
Step 5	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# service-policy output policy1`	Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.

Step 6	**end**
	or
	commit
	Example:
	`RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# end`
	or
	`RP/0/0RP0RSP0/CPU0:router(config-fr-vc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 7	**configure** **interface serial** *interface-path-id.subinterface* **pvc** *dlci* **encap** [**cisco** | **ietf**] **commit** **Example:** RP/0/0RP0RSP0/CPU0:router# configure RP/0/0RP0RSP0/CPU0:router (config)# interface serial 0/1/0/1.1 RP/0/0RP0RSP0/CPU0:router (config-subif)# pvc 20 RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# encap cisco RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# commit	Repeat Step 1 through Step 6 to bring up the serial subinterface and any associated PVC at the other end of the connection. **Note** The configuration on both ends of the subinterface connection must match.
Step 8	**show frame-relay pvc** *dlci-number* **Example:** RP/0/0RP0RSP0/CPU0:router# show frame-relay pvc 20	(Optional) Verifies the configuration of specified serial interface.
Step 9	**show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**} or **show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** | **output**} **Example:** RP/0/0RP0RSP0/CPU0:router# show policy-map interface serial 0/1/0/0.1 output or RP/0/0RP0RSP0/CPU0:router# show policy-map type qos interface serial 0/1/0/0.1 output	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface.

What to Do Next

- To modify the default Frame Relay configuration on serial interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" section of the *"Configuring Frame Relay on the Cisco ASR 9000 Series Router"* module in this manual.

Modifying the Keepalive Interval on Serial Interfaces

Perform this task to modify the keepalive interval on serial interfaces that have Cisco HDLC or PPP encapsulation enabled.

Note When you enable Cisco HDLC or PPP encapsulation on a serial interface, the default keepalive interval is 10 seconds. Use this procedure to modify that default keepalive interval.

Note Cisco HDLC is enabled by default on serial interfaces.

Prerequisites

Before modifying the keepalive timer configuration, ensure that Cisco HDLC or PPP encapsulation is enabled on the interface. Use the **encapsulation** command to enable Cisco HDLC or PPP encapsulation on the interface, as described in the "Configuring Optional Serial Interface Parameters" section on page 518.

Restrictions

Is there a comparable recommendation for ASR9k with keepalives for MDR upgrade? XR12k recommends disabling keepalives and CRS recommends setting interval to greater than or equal to 10 sec.

- Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend disabling keepalives on a Cisco XR 12000 Series Router.
- Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend configuring a keepalive interval of 10 seconds or more on a Cisco CRS-1 Router.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id*
3. **keepalive** {*seconds* | **disable**}
   or
   no keepalive
4. **end**
   or
   commit
5. **show interfaces** *type interface-path-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0RP0RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface serial`** *`interface-path-id`*
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/0`
		Specifies the serial interface name and notation *rack/slot/module/port* and enters interface configuration mode.

	Command or Action	Purpose

Step 3	**keepalive** {*seconds* | **disable**}
	or
	no keepalive
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config-if)# keepalive 3`
	or
	`RP/0/0RP0RSP0/CPU0:router(config-if)# no keepalive`
		Specifies the number of seconds between keepalive messages.
		• Use the **keepalive disable** command, the **no keepalive**, or the **keepalive** command with an argument of 0 to disable the keepalive feature.
		• The range is from 1 to 30 seconds. The default is 10 seconds.
		• If keepalives are configured on an interface, use the **no keepalive** command to disable the keepalive feature before configuring Frame Relay encapsulation on that interface.

Step 4	**end**
	or
	commit
	Example:
	`RP/0/0RP0RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/0RP0RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show interfaces serial** *interface-path-id* **Example:** `RP/0/0RP0RSP0/CPU0:router# show interfaces serial 0/1/0/0`	(Optional) Verifies the interface configuration.

How to Configure a Layer 2 Attachment Circuit

The Layer 2 AC configuration tasks are described in the following procedures:

- Creating a Serial Layer 2 Subinterface with a PVC
- Configuring Optional Serial Layer 2 PVC Parameters

Note After you configure an interface for Layer 2 switching, no routing commands such as **ipv4 address** are permissible. If any routing commands are configured on the interface, then the **l2transport** command is rejected.

Creating a Serial Layer 2 Subinterface with a PVC

The procedure in this section creates a Layer 2 subinterface with a PVC.

Prerequisites

Before you can create a subinterface on a serial interface, you must bring up a serial interface, as described in the "Bringing Up a Serial Interface" section on page 514.

Restrictions

Only one PVC can be configured for each serial subinterface.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **l2transport**
3. **pvc** *vpi/vci*
4. **end**
   or
   commit
5. Repeat Step 1 through Step 4 to bring up the serial subinterface and any associated PVC at the other end of the AC.

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0RP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface serial`** *`interface-path-id.subinterface`* **`l2transport`**
	Example:
	`RP/0/0RP0/CPU0:router(config)# interface serial 0/1/0/0.1 l2transport`
		Creates a subinterface and enters serial subinterface configuration mode for that subinterface.

	Command or Action	Purpose
Step 3	**pvc** *vpi*/*vci* **Example:** `RP/0/0RP0/CPU0:router(config-if)# pvc 5/20`	Creates a serial permanent virtual circuit (PVC) and enters serial Layer 2 transport PVC configuration mode. **Note** Only one PVC is allowed per subinterface.

Step 4	**end**
	or
	commit
	Example:
	`RP/0/0RP0/CPU0:router(config-fr-vc)# end`
	or
	`RP/0/0RP0/CPU0:router(config-fr-vc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	Repeat Step 1 through Step 4 to bring up the serial subinterface and any associated PVC at the other end of the AC.	Brings up the AC. **Note** The configuration on both ends of the AC must match.

What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional Serial Layer 2 PVC Parameters" section on page 531.
- For detailed information about configuring L2TPv3 in your network, see the "*Implementing Layer 2 Tunnel Protocol Version 3"* module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.* For detailed information about configuring L2VPNs, see the "*Implementing MPLS Layer 2 VPNs"* module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.*
- For detailed information about configuring L2TPv3 in your network, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software"* module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.* For detailed information about configuring L2VPNs, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software"* module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

Configuring Optional Serial Layer 2 PVC Parameters

This task describes the commands you can use to modify the default configuration on a serial Layer 2 PVC.

Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a Layer 2 subinterface, as described in the "Creating a Serial Layer 2 Subinterface with a PVC" section on page 529.

Restrictions

The configuration on both ends of the PVC must match for the connection to be active.

SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **fragment end-to-end** *fragment-size*
7. fragment-counter
8. **end**
   or
   commit
9. Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC.
10. **show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**}
    or
    show policy-map type qos interface serial *interface-path-id.subinterface* {**input** | **output**}

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0RP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface serial`** *`interface-path-id.subinterface`* **`l2transport`**
	Example:
	`RP/0/0RP0/CPU0:router(config)# interface serial 0/1/0/0.1 l2transport`
		Enters serial subinterface configuration mode for a Layer 2 serial subinterface.

	Command or Action	Purpose

Step 3	**`pvc`** *`dlci`*
	Example:
	`RP/0/0RP0/CPU0:router(config-if)# pvc 100`
		Enters serial Frame Relay PVC configuration mode for the specified PVC.

Step 4	**`encap`** `{`**`cisco`** `|` **`ietf`**`}`
	Example:
	`RP/0/0RP0/CPU0:router(config-fr-vc)#`
	`encapsulation aal5`
		Configures the encapsulation for a Frame Relay PVC.

Step 5	**`service-policy`** `{`**`input`** `|` **`output`**`}` *`policy-map`*
	Example:
	`RP/0/0RP0/CPU0:router (config-subif)#`
	`service-policy output policy1`
		Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.

Step 6	**`fragment end-to-end`** *`fragment-size`*
	Example:
	`RP/0/0/CPU0:router(config-fr-vc)# fragment`
	`end-to-end 100`
		Enables fragmentation of Frame Relay frames on an interface.
		Replace *fragment-size* with the number of payload bytes from the original Frame Relay frame that will go into each fragment. This number excludes the Frame Relay header of the original frame.
		On the Cisco 8-Port Channelized T1/E1 SPA, valid values are 128, 256, and 512.

Step 7	**`fragment-counter`**
	`Example:`
	`RP/0/0/CPU0:router(config-fr-vc)#`
	`fragment-counter`
		Enables fragmentation counters for a Frame Relay subinterface and PVC.

Step 8	**`end`**
	or
	`commit`
	Example:
	`RP/0/0RP0/CPU0:router(config-serial-l2transport`
	`-`
	`pvc)# end`
	or
	`RP/0/0RP0/CPU0:router(config-serial-l2transport`
	`-pvc)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

	Command or Action	Purpose
Step 9	Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC.	Brings up the AC. **Note** The configuration on both ends of the connection must match.

Step 10	**show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**}
	or
	show policy-map type qos interface serial *interface-path-id.subinterface* {**input** | **output**}
	Example:
	`RP/0/0RP0/CPU0:router# show policy-map interface pos 0/1/0/0.1 output`
	or
	`RP/0/0RP0/CPU0:router# show policy-map type qos interface pos 0/1/0/0.1 output`
		(Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface.

What to Do Next

- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "*Implementing Layer 2 Tunnel Protocol Version 3*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.*
- To configure an L2VPN, see the "*Implementing MPLS Layer 2 VPNs*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco CRS Router.*
- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*
- To configure an L2VPN, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

Configuring IPHC

This section contains the following step procedures:

Prerequisites for Configuring IPHC

IP header compression (IPHC) is supported on the following cards:

- Cisco 1-Port Channelized OC-12/STM-4

- Cisco 1-Port Channelized OC-48/STM-16
- Cisco 1-Port Channelized STM-1/OC-3
- Cisco 8-Port Channelized T1/E1 SPA
- Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA
- Cisco 2-Port and 4-Port Channelized T3 SPA
- Cisco Multirate 10G IP Services Engines SIPs
  - Cisco 12000-SIP-600
  - Cisco 12000-SIP-401
  - Cisco 12000-SIP-501
  - Cisco 12000-SIP-601
- SIP 700 SPA Interface Processor
- Cisco 2-Port Channelized OC-12c/DS0 SPA
- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 4-Port Channelized T3/DS0 SPA
- Cisco 8-Port Channelized T1/E1 SPA
- Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA

Configuring the IPHC Slot Level Command

This section describes how to configure the IP header compression (IPHC) slot level command, which reserves the IPHC resources, enables IPHC on the line card, and defines the maximum number of TCP and non-TCP connections for the nodes. This configuration must be done before an IPHC profile can be created.

Note IPHC slot level configuration is required on both the peer routers.

SUMMARY STEPS

To configure the IP header compression (IPHC) slot level, perform the following steps.

1. **config**
2. **iphc tcp connections** *max-number* **location** *node-id*
3. **iphc non-tcp connections** *max-number* **location** *node-id*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`config`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`iphc tcp connections`** *`max-number`* **`location`** *`node-id`*
	Example:
	`RP/0/0/CPU0:router(config)# iphc tcp connections 2000 location 0/1/cpu0`
		Sets the maximum number of TCP connections that may be configured for IPHC on a line card.
		The range is 1 to 2000.

Step 3	**`iphc non-tcp connections`** *`max-number`* **`location`** *`node-id`*
	Example:
	`RP/0/0/CPU0:router(config)# iphc non-tcp connections 20000 location 0/1/cpu0`
		Sets the maximum number of non-TCP connections that may be configured for IPHC on a line card.
		The range is 1 to 20000.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring an IPHC Profile

This section describes how to create and configure an IP header compression (IPHC) profile. This procedure is for TCP and non-TCP compression.

SUMMARY STEPS

To configure an IP header compression (IPHC) profile, perform the following steps.

1. **configure**
2. **iphc profile** *profile-name* **type** {**ietf** | **iphc**}
3. **tcp compression**
4. **tcp context absolute** *number-of-contexts*
5. **non-tcp compression**
6. **non-tcp context absolute** *number-of-contexts*
7. **rtp**
8. **refresh max-period** {*max-number* | ***infinite***}
9. **refresh rtp**
10. **feedback disable**
11. **max-header** *number-of-bytes*
12. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**iphc profile** *profile-name* **type** {**ietf** | **iphc**}
	Example:
	`RP/0/0/CPU0:router(config)# iphc profile Profile_1 type iphc`
		Creates an IPHC profile, sets the compression format type. and enters the IPHC profile configuration mode.

Step 3	**tcp compression** **Example:** `RP/0/0/CPU0:router(config-iphc-profile)# tcp compression`	Enables TCP compression in an IPHC profile.

	Command or Action	Purpose
Step 4	**tcp context absolute** number-of-contexts **Example:** RP/0/0/CPU0:router(config-iphc-profile)# tcp context absolute 255	Configures the maximum number of TCP contexts that are allowed for IPHC on a line card.
Step 5	**non-tcp compression** **Example:** RP/0/0/CPU0:router(config-iphc-profile)# non-tcp compression	Enables non-TCP compression in an IPHC profile.
Step 6	**non-tcp context absolute** number-of-contexts **Example:** RP/0/0/CPU0:router(config-iphc-profile)# non-tcp context absolute 255	Configures the maximum number of non-TCP contexts that are allowed for IPHC on a line card.
Step 7	**rtp** **Example:** RP/0/0/CPU0:router(config-iphc-profile)# rtp	Configures Real Time Protocol (RTP) on the interface.
Step 8	**refresh max-period** {*max-number* | **infinite**} **Example:** RP/0/0/CPU0:router(config-iphc-profile)# refresh max-period 50	Configures the maximum number of compressed IP header packets that are exchanged on a link before the IPHC context is refreshed.
Step 9	**refresh rtp** **Example:** RP/0/0/CPU0:router(config-iphc-profile)# refresh rtp	Enables the configured context refresh settings for RTP packets.
Step 10	**feedback disable** **Example:** RP/0/0/CPU0:router(config-iphc-profile)# feedback disable	Disables the IPHC context status feedback messages on an interface.

	Command or Action	Purpose

Step 11	`max-header number-of-bytes`
	Example:
	`RP/0/0/CPU0:router(config-iphc-profile)# max-header 20`
		Configures the maximum size (in bytes) of a compressed IP header.

Step 12	`end`
	or
	`commit`
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring an IPHC Profile

This section describes how to create and configure an IP header compression (IPHC) profile. This procedure is for TCP and non-TCP compression.

SUMMARY STEPS

To configure an IP header compression (IPHC) profile, perform the following steps.

1. **configure**
2. **iphc profile** *profile-name* **type** {**cisco** | **ietf** | **iphc**}
3. **tcp compression**
4. **tcp context absolute** *number-of-contexts*
5. **non-tcp compression**
6. **non-tcp context absolute** *number-of-contexts*
7. **rtp**
8. **refresh max-period** {*max-number* | **infinite**}
9. **refresh max-time** {*max-time* | **infinite**}
10. **refresh rtp**

11. **feedback disable**
12. **max-header** *number-of-bytes*
13. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**config** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**iphc profile** *profile-name* **type** {**cisco** | **ietf** | **iphc**}
	Example:
	`RP/0/RSP0/CPU0:router(config)# iphc profile Profile_1 type iphc`
		Creates an IPHC profile, sets the compression format type, and enters the IPHC profile configuration mode.

Step 3	**tcp compression** **Example:** `RP/0/RSP0/CPU0:router(config-iphc-profile)# tcp compression`	Enables TCP compression in an IPHC profile.
Step 4	**tcp context absolute** number-of-contexts **Example:** `RP/0/RSP0/CPU0:router(config-iphc-profile)# tcp context absolute 255`	Configures the maximum number of TCP contexts that are allowed for IPHC on a line card.
Step 5	**non-tcp compression** **Example:** `RP/0/RSP0/CPU0:router(config-iphc-profile)# non-tcp compression`	Enables non-TCP compression in an IPHC profile.
Step 6	**non-tcp context absolute** number-of-contexts **Example:** `RP/0/RSP0/CPU0:router(config-iphc-profile)# non-tcp context absolute 255`	Configures the maximum number of non-TCP contexts that are allowed for IPHC on a line card.
Step 7	**rtp** **Example:** `RP/0/RSP0/CPU0:router(config-iphc-profile)# rtp`	Configures Real Time Protocol (RTP) on the interface.

	Command or Action	Purpose

Step 8	**`refresh max-period`** {*`max-number`* | **`infinite`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-iphc-profile)# refresh max-period 50`
		Configures the maximum number of compressed IP header packets that are exchanged on a link before the IPHC context is refreshed.

Step 9	**`refresh max-time`** {*`max-time`* | **`infinite`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-iphc-profile)# refresh max-time 10`
		Configures the maximum time between context refreshes.

Step 10	**`refresh rtp`**
	Example:
	`RP/0/RSP0/CPU0:router(config-iphc-profile)# refresh rtp`
		Enables the configured context refresh settings for RTP packets.

Step 11	**`feedback disable`**
	Example:
	`RP/0/RSP0/CPU0:router(config-iphc-profile)# feedback disable`
		Disables the IPHC context status feedback messages on an interface.

Step 12	**`max-header`** *`number-of-bytes`*
	Example:
	`RP/0/RSP0/CPU0:router(config-iphc-profile)# max-header 20`
		Configures the maximum size (in bytes) of a compressed IP header.

Step 13	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Enabling an IPHC Profile on an Interface

This section describes how to enable an IP header compression (IPHC) profile on an interface by attaching the profile directly to the interface.

SUMMARY STEPS

To configure to enable an IPHC profile on an interface, perform the following steps.

1. **config**
2. **interface** *type interface-path-id*
3. **encapsulation ppp**
4. **ipv4 iphc profile** *profile-name* [**mode service-policy**]
5. **service policy input | output | type** *service-policy-name*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**config** **Example:** `RP/0/0RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface type** *interface-path-id* **Example:** `RP/0/0RSP0/CPU0:router(config)# interface serial 0/1/0/1`	Specifies the interface. **Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router. For more information about the syntax for the router, use the question mark (**?**) online help function.

Step 3	**encapsulation** {**hdlc** | **ppp** | **frame-relay** | **mfr**}
	Example:
	`RP/0/0RSP0/CPU0:router(config-if)# encapsulation ppp`
		Specifies Layer 2 encapsulation for the interface.

Step 4	**ipv4 iphc profile** *profile-name* [**mode service-policy**]
	Example:
	`RP/0/0RSP0/CPU0:router(config-if)# ipv4 iphc profile Profile_1`
	or
	`RP/0/0RSP0/CPU0:router(config-if)# ipv4 iphc profile Profile_1 mode service-policy`
		Attaches an IPHC profile to the interface:
		• *profile-name*—Text name of the IPHC profile to attach to the interface.
		• **mode service-policy**—Specifies that the IPHC profile applies only to a QoS service policy.

	Command or Action	Purpose
Step 5	**service policy output** *service-policy-name* **Example:** `RP/0/0RSP0/CPU0:router(config-if)# service policy input | output | type service-policy-name`	(Optional) Specifies the name of the QoS service policy to which the IPHC profile applies. Only output service policies are allowed. Used only when **mode service-policy** is specified in Step 2.

Step 6	**end**
	or
	commit
	Example:
	`RP/0/0RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/0RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuration Examples for Serial Interfaces

This section provides the following configuration examples:

- Bringing Up and Configuring a Serial Interface with Cisco HDLC Encapsulation: Example, page 542
- Configuring a Serial Interface with Frame Relay Encapsulation: Example, page 543
- Configuring a Serial Interface with PPP Encapsulation: Example, page 545
- IPHC Configuration: Examples, page 545

Bringing Up and Configuring a Serial Interface with Cisco HDLC Encapsulation: Example

The following example shows how to bring up a basic serial interface with Cisco HDLC encapsulation:

```
RP/0/0RP0RSP0/CPU0:Router#config
RP/0/0RP0RSP0/CPU0:Router(config)# interface serial 0/3/0/0/0:0
RP/0/0RP0RSP0/CPU0:Router(config-if)# ipv4 address 192.0.2.2 255.255.255.252
RP/0/0RP0RSP0/CPU0:Router(config-if)# no shutdown
```

```
RP/0/0RP0RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure the interval between keepalive messages to be 10 seconds:

```
RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0RP0RSP0/CPU0:router(config-if)# keepalive 10
RP/0/0RP0RSP0/CPU0:router(config-if)# commit
```

The following example shows how to modify the optional serial interface parameters:

```
RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0RP0RSP0/CPU0:Router(config-if)# serial
RP/0/0RP0RSP0/CPU0:Router(config-if-serial)# crc 16
RP/0/0RP0RSP0/CPU0:Router(config-if-serial)# invert
RP/0/0RP0RSP0/CPU0:Router(config-if-serial)# scramble
RP/0/0RP0RSP0/CPU0:Router(config-if-serial)# transmit-delay 3
RP/0/0RP0RSP0/CPU0:Router(config-if-serial)# commit
```

The following is sample output from the **show interfaces serial** command:

```
RP/0/0RP0RSP0/CPU0:Router# show interfaces serial 0/0/3/0/5:23
Serial0/0/3/0/5:23 is down, line protocol is down
  Hardware is Serial network interface(s)
  Internet address is Unknown
  MTU 1504 bytes, BW 64 Kbit
     reliability 143/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, crc 16,  loopback not set, keepalive set (10 sec)
  Last clearing of "show interface" counters 18:11:15
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2764 packets input, 2816 bytes, 3046 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     3046 input errors, 1 CRC, 0 frame, 0 overrun, 2764 ignored, 281 abort
     2764 packets output, 60804 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

Configuring a Serial Interface with Frame Relay Encapsulation: Example

The following example shows how to create a serial interface on a SPA with Frame Relay encapsulation and a serial subinterface with a PVC on router 1:

```
RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/0
RP/0/0RP0RSP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0RP0RSP0/CPU0:router(config-if)#frame-relay intf-type dce
RP/0/0RP0RSP0/CPU0:router(config-if)# no shutdown
RP/0/0RP0RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/0.1 point-to-point
```

```
RP/0/0RP0RSP0/CPU0:router (config-subif)#ipv4 address 10.20.3.1/24
RP/0/0RP0RSP0/CPU0:router (config-subif)# pvc 16
RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# encapsulation ietf
RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# commit
RP/0/0RP0RSP0/CPU0:router(config-fr-vc)# exit
RP/0/0RP0RSP0/CPU0:router(config-subif)# exit
RP/0/0RP0RSP0/CPU0:router(config)# exit

RP/0/0RP0RSP0/CPU0:router# show interface serial 0/1/0/0
Wed Oct  8 04:14:39.946 PST DST
Serial0/1/0/0 is up, line protocol is up
  Interface state transitions: 5
  Hardware is Serial network interface(s)
  Internet address is 10.20.3.1/24
  MTU 4474 bytes, BW 44210 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 16,
  Scrambling is disabled, Invert data is disabled
  LMI enq sent  0, LMI stat recvd 0, LMI upd recvd 0
  LMI enq recvd 880, LMI stat sent  880, LMI upd sent  0 , DCE LMI up
  LMI DLCI 1023  LMI type is CISCO  frame relay DCE
  Last clearing of "show interface" counters 02:23:04
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     858 packets input, 11154 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     858 packets output, 12226 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how to create a serial interface on a SPA with Frame Relay encapsulation and a serial subinterface with a PVC on router 2, which is connected to router 1:

```
RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/1
RP/0/0RP0RSP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0RP0RSP0/CPU0:router(config-if)# no shutdown
RP/0/0RP0RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/1/0/1.1 point-to-point
RP/0/0RP0RSP0/CPU0:router (config-subif)#ipv4 address 10.20.3.2/24
RP/0/0RP0RSP0/CPU0:router (config-subif)# pvc 16
RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# encapsulation ietf
RP/0/0RP0RSP0/CPU0:router (config-fr-vc)# commit
```

```
RP/0/0RP0RSP0/CPU0:router(config-fr-vc)# exit
RP/0/0RP0RSP0/CPU0:router(config-subif)# exit
RP/0/0RP0RSP0/CPU0:router(config)# exit

RP/0/0RP0RSP0/CPU0:router# show interface serial 0/1/0/1
Wed Oct  8 04:13:45.046 PST DST
Serial0/1/0/1 is up, line protocol is up
  Interface state transitions: 7
  Hardware is Serial network interface(s)
  Internet address is Unknown
  MTU 4474 bytes, BW 44210 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 16,
  Scrambling is disabled, Invert data is disabled
  LMI enq sent  1110, LMI stat recvd 875, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  Last clearing of "show interface" counters 02:22:09
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     853 packets input, 12153 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     853 packets output, 11089 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

Configuring a Serial Interface with PPP Encapsulation: Example

The following example shows how to create and configure a serial interface with PPP encapsulation:

```
RP/0/0RP0RSP0/CPU0:router# configure
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0RP0RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0RP0RSP0/CPU0:router(config-if)# encapsulation ppp
RP/0/0RP0RSP0/CPU0:router(config-if)# no shutdown
RP/0/0RP0RSP0/CPU0:router(config-if)# ppp authentication chap MIS-access
RP/0/0RP0RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure serial interface 0/3/0/0/0:0 to allow two additional retries after an initial authentication failure (for a total of three failed authentication attempts):

```
RP/0/0RP0RSP0/CPU0:router# configuration
RP/0/0RP0RSP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0RP0RSP0/CPU0:router(config-if)# encapsulation ppp
RP/0/0RP0RSP0/CPU0:router(config-if)# ppp authentication chap
RP/0/0RP0RSP0/CPU0:router(config-if)# ppp max-bad-auth 3
RP/0/0RP0RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

IPHC Configuration: Examples

This section provides the following examples:

- IPHC Profile Configuration: Example, page 546
- IPHC on a Serial Interface Configuration: Examples, page 546
- IPHC on Multilink Configuration: Example, page 546

- IPHC on a Serial Interface with MLPPP/LFI and QoS Configuration: Example, page 547

IPHC Profile Configuration: Example

The following example shows how to configure an IPHC Profile:

```
config
  iphc tcp connections 6000 location 0/2/1
  iphc non-tcp connections 6000 location 0/2/1
  iphc profile Profile_1 type iphc
    tcp compression
    tcp context absolute 255
    non-tcp compression
    non-tcp context absolute 255
    rtp
    refresh max-period 50
    refresh max-time 10
    refresh rtp
    feedback disable
    max-header 20
commit
```

IPHC on a Serial Interface Configuration: Examples

Example 1

The following example shows how to enable an IP header compression (IPHC) profile on a serial interface by attaching the profile directly to the interface:

```
config
  interface serial 0/1/0/1
    encapsulation ppp
    ipv4 iphc profile Profile_1
commit
```

Example 2

The following example shows how to enable an IP header compression (IPHC) profile on an interface by specifying a QoS service policy that contains an IPHC profile:

```
config
  interface serial 0/1/0/1:1
    encapsulation ppp
    ipv4 iphc profile Profile_2 mode service-policy
    service-policy output ip_header_compression_policy_map
commit
```

IPHC on Multilink Configuration: Example

The following example shows how to configure an IP header compression (IPHC) on a multilink interface:

```
config
  interface multilink 0/4/3/0/4
    ipv4 address 10.10.10.10
    encapsulation ppp
    ipv4 iphc profile Profile_1
    commit
  interface serial 0/1/0/1:1
    encapsulation ppp
```

```
        multilink group 4
        commit
```

IPHC on a Serial Interface with MLPPP/LFI and QoS Configuration: Example

The following example shows how to configure IP header compression (IPHC) on a serial interface with LFI and by specifying a QoS service policy that contains an IPHC profile:

```
config
  interface multilink 0/4/3/0/4
    ipv4 address 10.10.10.10
      multilink
        fragment-size 128
        interleave
    ipv4 iphc profile Profile_2 mode service-policy
    service-policy output SP_2
    commit
  interface serial 0/1/0/1:2
    encapsulation ppp
    multilink group 4
    commit
```

Additional References

The following sections provide references related to T3/E3 and T1/E1 controllers and serial interfaces.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using Cisco IOS XR software	*Cisco IOS XR Getting Started Guide*
Cisco IOS XR AAA services configuration information	*Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference*
Information about configuring interfaces and other components on the Cisco CRS-1 Router from a remote Craft Works Interface (CWI) client management application	*Cisco Craft Works Interface Configuration Guide*

Standards

Standards	Title
FRF.1.2	*PVC User-to-Network Interface (UNI) Implementation Agreement - July 2000*
ANSI T1.617 Annex D	—
ITU Q.933 Annex A	—

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1294	*Multiprotocol Interconnect Over Frame Relay*
RFC 1315	*Management Information Base for Frame Relay DTEs*
RFC 1490	*Multiprotocol Interconnect Over Frame Relay*
RFC 1586	*Guidelines for Running OSPF Over Frame Relay Networks*
RFC 1604	*Definitions of Managed Objects for Frame Relay Service*
RFC 2115	*Management Information Base for Frame Relay DTEs Using SMIv2*
RFC 2390	*Inverse Address Resolution Protocol*
RFC 2427	*Multiprotocol Interconnect Over Frame Relay*
RFC 2954	*Definitions of Managed Objects for Frame Relay Service*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Frame Relay on the Cisco ASR 9000 Series Router

This module describes the optional configurable Frame Relay parameters available on Packet-over-SONET/SDH (POS), multilink, and serial interfaces configured with Frame Relay encapsulation.

Feature History for Configuring Frame Relay Interfaces on Cisco IOS XR Software

Release	Modification
Release 4.0.0	Support for Frame Relay was added for the following SPAs: • Cisco 2-Port Channelized OC-12c/DS0 SPA • Cisco 1-Port Channelized OC-48/STM-16 SPA • Cisco 8-Port OC-12c/STM-4 POS SPA • Cisco 2-Port OC-48c/STM-16 POS/RPR SPA • Cisco 1-Port OC-192c/STM-64 POS/RPR XFP SPA Support for these Frame Relay features was added for the Cisco 2-Port Channelized OC-12c/DSO SPA: • Multilink Frame Relay (FRF.16) • End-to-End Fragmentation (FRF.12)
Release 4.0.1	Support for Frame Relay was added for these SPAs: • Cisco 1-Port Channelized OC-3/STM-1 SPA • Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA • Cisco 4-Port OC-3c/STM-1 POS SPA • Cisco 8-Port OC-3c/STM-1 POS SPA
Release 4.1.0	Support for Frame Relay was added for these SPAs: • Cisco 4-Port Channelized T3/DS0 SPA • Cisco 8-Port Channelized T1/E1 SPA

Contents

Prerequisites for Configuring Frame Relay

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring Frame Relay, be sure that the following conditions are met:

- Your hardware must support POS or serial interfaces.
- You have enabled Frame Relay encapsulation on your interface with the **encapsulation frame relay** command, as described in the appropriate module:
  - To enable Frame Relay encapsulation on a multilink bundle interface, see the "Configuring Multilink Frame Relay Bundle Interfaces" section on page 562.
  - To enable Frame Relay encapsulation on a POS interface, see the *"Configuring POS Interfaces onthe Cisco ASR 9000 Series Router"* module in this manual.
  - To enable Frame Relay encapsulation on a serial interface, see the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module in this manual.

Information About Frame Relay Interfaces

The following sections explain the various aspects of configuring Frame Relay interfaces:

- Frame Relay Encapsulation, page 550
- Multilink Frame Relay (FRF.16), page 553
- End-to-End Fragmentation (FRF.12), page 557

Frame Relay Encapsulation

On the Cisco ASR 9000 Series Router, Frame Relay is supported on POS and serial main interfaces, and on PVCs that are configured under those interfaces. To enable Frame Relay encapsulation on an interface, use the **encapsulation frame-relay** command in interface configuration mode.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (this is the default)
- IETF

Use the **encapsulation frame-relay** command in interface configuration mode to configure Cisco or IETF encapsulation on a PVC.

Note If the encapsulation type is not configured explicitly for a PVC with the **encapsulation** command, then that PVC inherits the encapsulation type from the main interface.

The **encapsulation frame relay** and **encap (PVC)** commands are described in the following modules:

- To enable Frame Relay encapsulation on a POS interface, see the *"Configuring POS Interfaces onthe Cisco ASR 9000 Series Router"* module in this manual.
- To enable Frame Relay encapsulation on a serial interface, see the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module in this manual.

When an interface is configured with Frame Relay encapsulation and no additional configuration commands are applied, the default interface settings shown in Table 23 are present. These default settings can be changed by configuration as described in this module.

Table 23 Frame Relay Encapsulation Default Settings

Parameter	Configuration File Entry	Default Settings	Command Mode
PVC Encapsulation	**encap** {**cisco** | **ietf**}	**cisco** **Note** When the **encap** command is not configured, the PVC encapsulation type is inherited from the Frame Relay main interface.	PVC configuration
Type of support provided by the interface	**frame-relay intf-type** {**dce** | **dte**}	**dte**	Interface configuration
LMI type supported on the interface	**frame-relay lmi-type** [**ansi** | **cisco** | **q933a**]	For a DCE, the default setting is **cisco**. For a DTE, the default setting is synchronized to match the LMI type supported on the DCE. **Note** To return an interface to its default LMI type, use the **no frame-relay lmi-type** [**ansi** | **cisco** | **q933a**] command.	Interface configuration
Disable or enable LMI	**frame-relay lmi disable**	LMI is enabled by default on Frame Relay interfaces. To reenable LMI on an interface after it has been disabled, use the **no frame-relay lmi disable** command.	Interface configuration

Note The default settings of LMI polling-related commands appear in Table 24 on page 552 and Table 25 on page 553.

LMI

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay User-Network Interface (UNI).

Frame Relay interfaces supports the following types of LMI on UNI interfaces:

- ANSI—ANSI T1.617 Annex D
- Q.933—ITU-T Q.933 Annex A
- Cisco

Use the **frame-relay lmi-type** command to configure the LMI type to be used on an interface.

Note The LMI type that you use must correspond to the PVCs configured on the main interface. The LMI type must match on both ends of a Frame Relay connection.

If your router functions as a switch connected to another non-Frame Relay router, use the **frame-relay intf-type dce** command to configure the LMI type to support data communication equipment (DCE).

If your router is connected to a Frame Relay network, use the **frame-relay intf-type dte** command to configure the LMI type to support data terminal equipment (DTE).

Note LMI type auto-sensing is supported on DTE interfaces by default.

Use the **show frame-relay lmi** and **show frame-relay lmi-info** commands in EXEC mode to display information and statistics for the Frame Relay interfaces in your system. (When specifying the *type* and *interface-path-id* arguments, you must specify information for the main interface.) You can modify the error threshold, event count, and polling verification timer and then use the **show frame-relay lmi** command to gather information that can help you monitor and troubleshoot Frame Relay interfaces.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA :

```
(MTU - 13)/8 = maximum number of PVCs
```

The default number of PVCs supported on POS PVCs configured with **cisco** LMI is 557, while the default number of PVCs supported on serial PVCs configured with **cisco** LMI is 186.

For LMI types that are not from Cisco, up to 992 PVCs are supported under a single main interface.

Note If a specific LMI type is configured on an interface, use the **no frame-relay lmi-type** [**ansi** | **cisco** | **q933a**] command to bring the interface back to the default LMI type.

Table 24 describes the commands that can be used to modify LMI polling options on PVCs configured for a DCE.

Table 24 LMI Polling Configuration Commands for DCE

Parameter	Configuration File Entry	Default Settings
Sets the error threshold on a DCE interface.	**lmi-n392dce** *threshold*	**3**
Sets the monitored event count.	**lmi-n393dce** *events*	**4**
Sets the polling verification timer on the DCE end.	**lmi-t392dce** *seconds*	**15**

Table 25 describes the commands that can be used to modify LMI polling options on PVCs configured for a DTE.

Table 25 LMI Polling Configuration Commands for DTE

Parameter	Configuration File Entry	Default Settings
Set the number of Line Integrity Verification (LIV) exchanges performed before requesting a full status message.	**lmi-n391dte** *polling-cycles*	**6**
Sets the error threshold.	**lmi-n392dte** *threshold*	**3**
Sets the monitored event count.	**lmi-n393dte** *events*	**4**
Sets the polling interval (in seconds) between each status inquiry from the DTE end.	**frame-relay lmi-t391dte** *seconds*	**10**

Multilink Frame Relay (FRF.16)

Multilink Frame Relay (MFR) is supported only on the following shared port adapters (SPAs):

- Cisco 1-Port Channelized STM-1/OC-3 SPA
- Cisco 2-Port Channelized OC-12c/DSO SPA

Multilink Frame Relay High Availability

MFR supports the following levels of high availability support:

- MFR supports a process restart, but some statistics will be reset during a restart of certain processes.
- MFR member links remain operational during a route switch processor (RSP) switchover.

Multilink Frame Relay Configuration Overview

A multilink Frame Relay interface is part of a multilink bundle that allows Frame Relay encapsulation on its interfaces. You create a multilink Frame Relay interface by configuring the following components:

- MgmtMultilink controller
- Multilink bundle interface that allows Frame Relay encapsulation
- Bundle identifier name
- Multilink Frame Relay subinterfaces
- Bundle interface bandwidth class
- Serial interfaces

MgmtMultilink Controller

You configure a multilink bundle under a controller, using the following commands:

controller MgmtMultilink *rack/slot/bay/controller-id*

bundle *bundleId*

This configuration creates the controller for a generic multilink bundle. The controller ID number is the zero-based index of the controller chip. Currently, the SPAs that support multilink Frame Relay have only one controller per bay; therefore, the controller ID number is always zero (0).

Multilink Bundle Interface

After you create the multilink bundle, you create a multilink bundle interface that allows Frame Relay encapsulation, using the following commands:

interface multilink *interface-path-id*

encapsulation frame-relay

This configuration allows you to create multilink Frame Relay subinterfaces under the multilink bundle interface.

Note After you set the encapsulation on a multilink bundle interface to Frame Relay, you cannot change the encapsulation if the interface has member links or any member links associated with a multilink bundle.

Bundle Identifier Name

Note Bundle identifier name is configurable only under Frame Relay Forum 16.1 (FRF 16.1).

The bundle identifier (**bid**) name value identifies the bundle interface at both endpoints of the interface. The bundle identifier name is exchanged in the information elements to ensure consistent link assignments.

By default, the interface name, for example, Multilink 0/4/1/0/1, is used as the bundle identifier name. However, you can optionally create a name using the **frame-relay multilink bid** command.

Note Regardless of whether you use the default name or create a name using the frame-relay multilink bid command, it is recommended that each bundle have a unique name.

The bundle identifier name can be up to 50 characters including the null termination character. The bundle identifier name is configured at the bundle interface level and is applied to each member link.

You configure the bundle identifier name using the following commands:

interface multilink *interface-path-id*

frame-relay multilink bid *bundle-id-name*

Multilink Frame Relay Subinterfaces

You configure a multilink Frame Relay subinterface, using the following command:

interface multilink *interface-path-id*[.*subinterface* {**l2transport** | **point-to-point**}]

You can configure up to 992 subinterfaces on a multilink bundle interface.

Note You configure specific Frame Relay interface features at the subinterface level.

Multilink Frame-Relay Subinterface Features

The following commands are available to set specific features on a multilink Frame Relay bundle subinterface:

- **mtu** *MTU size*
- **description**
- **shutdown**
- **bandwidth** *bandwidth*
- **service-policy** {**input** | **output**} *policymap-name*

Note When entering the **service-policy** command, which enables you to attach a policy map to a multilink Frame Relay bundle subinterface, you must do so while in Frame Relay PVC configuration mode. For more information, see Configuring Multilink Frame Relay Bundle Interfaces, page 562.

Bundle Interface Bandwidth Class

Note Bandwidth class is configurable only under a multilink bundle interface.

You can configure one of three types of bandwidth classes on a multilink Frame Relay interface:

- a—Bandwidth Class A
- b—Bandwidth Class B
- c—Bandwidth Class C

When Bandwidth Class A is configured and one or more member links are up (PH_ACTIVE), the bundle interface is also up and BL_ACTIVATE is signaled to the Frame Relay connections. When all the member links are down, the bundle interface is down and BL_DEACTIVATE is signaled to the Frame Relay connections.

When Bandwidth Class B is configured and all the member links are up (PH_ACTIVE), the bundle interface is up and BL_ACTIVATE is signaled to the Frame Relay connections. When any member link is down, the bundle interface is down and BL_ACTIVATE is signaled to the Frame Relay connections.

When Bandwidth Class C is configured, you must also set the bundle link threshold to a value between 1 and 255. The threshold value is the minimum number of links that must be up (PH_ACTIVE) for the bundle interface to be up and for BL_ACTIVATE to be signaled to the Frame Relay connections. When the number of links that are up falls below this threshold, the bundle interface goes down and BL_DEACTIVATE is signaled to the Frame Relay connections. When 1 is entered as the threshold value, the behavior is identical to Bandwidth Class A. If you enter a threshold value that is greater than the number of member links that are up, the bundle remains down.

You configure the bandwidth class for a Frame Relay multilink bundle interface using the following commands:

interface multilink *interface-path-id*

frame-relay multilink bandwidth-class {**a** | **b** | **c** [*threshold*]}

The default is a (Bandwidth Class A).

Serial Interfaces

After the T3 and T1 controllers are configured, you can add serial interfaces to the multilink Frame Relay bundle subinterface by configuring the serial interface, encapsulating it as multilink Frame Relay (mfr), assigning it to the bundle interface (specified by the multilink group number), and configuring a name for the link. You may also configure MFR acknowledge timeout value, retry count for retransmissions and hello interval, for the bundle link.

You configure a multilink Frame Relay serial interface using the following commands:

interface serial *rack/slot/module/port/t1-num:channel-group-number*

encapsulation mfr

multilink group *group number*

frame-relay multilink lid *link-id name*

frame-relay multilink ack *ack-timeout*

frame-relay multilink hello *hello-interval*

frame-relay multilink retry *retry-count*

Note All serial links in an MFR bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MFR bundle. The Cisco IOS XR software blocks attempts to configure a serial interface as a member of an MFR bundle if the interface is configured with a nondefault MTU value as well as attempts to change the **mtu** command value for a serial interface that is configured as a member of an MFR bundle.

Show Commands

You can verify a multilink Frame Relay serial interface configuration using the following **show** commands:

show frame-relay multilink location *node id*

show frame-relay multilink interface serial *interface-path-id* [**detail** | **verbose**]

The following example shows the display output of the **show frame-relay multilink location** command:

```
RP/0/RSP0/CPU0:router# show frame-relay multilink  location  0/4/cpu0
Member interface: Serial0/4/2/0/9:0, ifhandle 0x05007b00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800

Bundle interface: Multilink0/4/2/0/2, ifhandle 0x05007800
    Member Links: 4 active, 0 inactive
    State = Up,   BW Class = C (threshold   3)
    Member Links:
    Serial0/4/2/0/12:0, HW state = Up, link state = Up
    Serial0/4/2/0/11:0, HW state = Up, link state = Up
    Serial0/4/2/0/10:0, HW state = Up, link state = Up
    Serial0/4/2/0/9:0, HW state = Up, link state = Up

Member interface: Serial0/4/2/0/10:0, ifhandle 0x05007c00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800

Member interface: Serial0/4/2/0/11:0, ifhandle 0x05007d00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800
Member interface: Serial0/4/2/0/12:0, ifhandle 0x05007e00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800
```

The following example shows the display output of

```
RP/0/RSP0/CPU0:router# show frame-relay multilink interface serial 0/4/2/0/10:0

Member interface: Serial0/4/2/0/10:0, ifhandle 0x05007c00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800
```

End-to-End Fragmentation (FRF.12)

You can configure an FRF.12 end-to-end fragmentation connection using the data-link connection identifier (DLCI). However, it must be done on a channelized Frame Relay serial interface.

Note The **fragment end-to-end** command is not allowed on Packet-over-SONET/SDH (POS) interfaces or under the DLCI of a multilink Frame Relay bundle interface.

You configure FRF.12 end-to-end fragmentation on a DLCI connection using the following command:

fragment end-to-end *fragment-size*

The *fragment-size* argument defines the size of the fragments, in bytes, for the serial interface.

Note On a DLCI connection, we highly recommend that you configure an egress service policy that classifies packets into high and low priorities, so that interleaving of high-priority and low-priority fragments occurs.

Configuring Frame Relay

The following sections describe how to configure Frame Relay interfaces.

- Modifying the Default Frame Relay Configuration on an Interface, page 557
- Disabling LMI on an Interface with Frame Relay Encapsulation, page 560
- Configuring Multilink Frame Relay Bundle Interfaces, page 562
- Configuring FRF.12 End-to-End Fragmentation on a Channelized Frame Relay Serial Interface, page 568

Modifying the Default Frame Relay Configuration on an Interface

Perform this task to modify the default Frame Relay parameters on a Packet-over-SONET/SDH (POS), multilink, or serial interface with Frame Relay encapsulation.

Prerequisites

Before you can modify the default Frame Relay configuration, you need to enable Frame Relay on the interface, as described in the following modules:

- To enable Frame Relay encapsulation on a POS interface, see the *"Configuring POS Interfaces onthe Cisco ASR 9000 Series Router"* module in this manual.
- To enable Frame Relay encapsulation on a serial interface, see the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module in this manual.

Note Before enabling Frame Relay encapsulation on a POS or serial interface, make certain that you have not previously assigned an IP address to the interface. If an IP address is assigned to the interface, you will not be able to enable Frame Relay encapsulation. For Frame Relay, the IP address and subnet mask are configured on the subinterface.

Restrictions

- The LMI type must match on both ends of the connection for the connection to be active.
- Before you can remove Frame Relay encapsulation on an interface and reconfigure that interface with PPP or HDLC encapsulation, you must remove all interfaces, subinterface, LMI, and Frame Relay configuration from that interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **frame-relay intf-type** {**dce** | **dte**}
4. **frame-relay lmi-type** [**ansi** | **cisco** | **q933a**]
5. **encap** {**cisco** | **ietf**}
6. **end**
or
commit
7. **show interfaces** [**summary** | [*type interface-path-id*] [**brief** | **description** | **detail** | **accounting** [**rates**]]] [**location** *node-id*]

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface pos 0/4/0/1`	Enters interface configuration mode.

Step 3	**frame-relay intf-type** {**dce** | **dte**}
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# frame-relay intf-type dce`
		Configures the type of support provided by the interface.
		• If your router functions as a switch connected to another router, use the **frame-relay intf-type dce** command to configure the LMI type to support data communication equipment (DCE).
		• If your router is connected to a Frame Relay network, use the **frame-relay intf-type dte** command to configure the LMI type to support data terminal equipment (DTE).
		Note The default interface type is DTE.

Step 4	**frame-relay lmi-type** [**ansi** | **q933a** | **cisco**] **Example:** `RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi-type ansi`	Selects the LMI type supported on the interface. • Enter the **frame-relay lmi-type ansi** command to use LMI as defined by ANSI T1.617a-1994 Annex D. • Enter the **frame-relay lmi-type cisco** command to use LMI as defined by Cisco (not standard). • Enter the **frame-relay lmi-type q933a** command to use LMI as defined by ITU-T Q.933 (02/2003) Annex A. **Note** The default LMI type is Cisco.

Step 5	**encap** {**cisco** | **ietf**}
	Example:
	`RP/0/RSP0/CPU0:router (config-fr-vc)# encap ietf`
		Configures the encapsulation for a Frame Relay PVC.
		Note If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main interface.

	Command or Action	Purpose

Step 6	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 7	**show interfaces** [**summary** | [*type interface-path-id*] [**brief** | **description** | **detail | accounting** [**rates**]]] [**location** *node-id*] **Example:** `RP/0/RSP0/CPU0:router# show interface pos 0/4/0/1`	(Optional) Verifies the configuration for the specified interface.

Disabling LMI on an Interface with Frame Relay Encapsulation

Perform this task to disable LMI on interfaces that have Frame Relay encapsulation.

Note LMI is enabled by default on interfaces that have Frame Relay encapsulation enabled. To reenable LMI on an interface after it has been disabled, use the **no frame-relay lmi disable** command in interface configuration mode.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **frame-relay lmi disable**
4. **end**
   or
   commit
5. **show interfaces** [**summary** | [*type interface-path-id*] [**brief** | **description** | **detail** | **accounting** [**rates**]]] [**location** *node-id*]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface POS 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`frame-relay lmi disable`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi disable`
		Disables LMI on the specified interface.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show interfaces** [**summary** | [*type interface-path-id*] [**brief** | **description** | **detail** | **accounting** [**rates**]]] [**location** *node-id*] **Example:** `RP/0/RSP0/CPU0:router# show interfaces POS 0/1/0/0`	(Optional) Verifies that LMI is disabled on the specified interface.

Configuring Multilink Frame Relay Bundle Interfaces

Perform these steps to configure a multilink Frame Relay (MFR) bundle interface and its subinterfaces.

Prerequisites

Before configuring MFR bundles, be sure you have the following SPA installed:

- 1-Port Channelized STM-1/OC-3 SPA
- 2-Port Channelized OC-12c/DS0 SPA

Restrictions

- All member links in a multilink Frame Relay bundle interface must be of the same type (for example, T1s or E1s). The member links must have the same framing type, such as point-to-point, and they must have the same bandwidth class.
- All member links must be full T1s or E1s. Fractional links, such as DS0s, are not supported.
- All member links must reside on the same SPA; otherwise, they are considered to be unrelated bundles.
- All member links must be connected to the same line card or SPA at the far end.
- A maximum of 992 MFR subinterfaces is supported on each main interface, based on the supported DLCI range 16–1007.
- The Cisco 1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12c/DS0 SPA have the following additional guidelines:
  - A maximum of 700 MFR bundles per line card is supported.
  - A maximum of 2600 MFR bundles per system is supported.
  - A maximum of 4000 Frame Relay Layer 3 subinterfaces per line card is supported.
  - A maximum of 8000 Frame Relay Layer 3 subinterfaces per system is supported.
- Fragmentation on a Frame Relay subinterface that is part of an MLFR bundle is not supported.
- All serial links in an MFR bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MFR bundle. The Cisco IOS XR software blocks the following:
  - Attempts to configure a serial interface as a member of an MFR bundle if the interface is configured with a nondefault MTU value.
  - Attempts to change the **mtu** command value for a serial interface that is configured as a member of an MFR bundle.

SUMMARY STEPS

1. **configure**
2. **controller MgmtMultilink** *rack/slot/bay/controller-id*
3. **exit**
4. **controller t3** *interface-path-id*
5. **mode** *type*

6. **clock source** {**internal** | **line**}
7. **exit**
8. **controller** {**t1** | **e1**} *interface-path-id*
9. **channel-group** *channel-group-number*
10. **timeslots** *range*
11. **exit**
12. **exit**
13. **interface multilink** *interface-path-id*[*.subinterface* {**l2transport** | **point-to-point**}]
14. **encapsulation frame-relay**
15. **frame-relay multilink bid** *bundle-id-name*
16. **frame-relay multilink bandwidth-class** {**a** | **b** | **c** [*threshold*]}
17. **exit**
18. **interface multilink** *interface-path-id*[*.subinterface* {**l2transport** | **point-to-point**}]
19. **ipv4 address** *ip-address*
20. **pvc** *dlci*
21. **service-policy** {**input** | **output**} *policy-map*
22. **exit**
23. **exit**
24. **interface serial** *interface-path-id*
25. **encapsulation mfr**
26. **multilink group** *group-id*
27. **frame-relay multilink lid** *link-id name*
28. **frame-relay multilink ack** *ack-timeout*
29. **frame-relay multilink hello** *hello-interval*
30. **frame-relay multilink retry** *retry-count*
31. **exit**
32. **end**
    or
    commit
33. **exit**
34. **show frame-relay multilink interface** *type interface-path-id* [**detail** | **verbose**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# config`
		Enters global configuration mode.

Step 2	**`controller MgmtMultilink`** *`rack/slot/bay/controller-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller MgmtMultilink 0/1/0/0`
		Creates the controller for a generic multilink bundle in the *rack/slot/bay/controller-id* notation and enters the multilink management configuration mode. The controller ID number is the zero-based index of the controller chip. Currently, the SPAs that support multilink Frame Relay have only one controller per bay; therefore, the controller ID number is always zero (0).

Step 3	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-mgmtmultilink)# exit`
		Exits the multilink management configuration mode.

Step 4	**`controller t3`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Specifies the T3 controller name in the *rack/slot/module/port* notation and enters T3 configuration mode.

Step 5	**`mode`** *`type`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# mode t1`
		Configures the type of multilinks to channelize; for example, 28 T1s.

Step 6	**`clock source`** {**`internal`** | **`line`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# clock source internal`
		(Optional) Sets the clocking for individual E3 links.
		Note The default clock source is **internal**.
		Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Step 7	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# exit`
		Exits T3/E3 controller configuration mode.

Step 8	**`controller`** {**`t1`** | **`e1`**} *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t1 0/1/0/0/0`
		Enters T1 or E1 configuration mode.

	Command or Action	Purpose
Step 9	**channel-group** *channel-group-number* **Example:** `RP/0/RSP0/CPU0:router(config-t1)# channel-group 0`	Creates a T1 channel group and enters channel group configuration mode for that channel group.
Step 10	**timeslots** *range* **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24`	Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group. • For T1 controllers—Range is from 1 to 24 time slots. • For E1 controllers—Range is from 1 to 31 time slots. • You can assign all time slots to a single channel group, or you can divide the time slots among several channel groups.
Step 11	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit`	Exits channel group configuration mode.
Step 12	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t1)# exit`	Exits T1 configuration mode.
Step 13	**interface multilink** *interface-path-id*[.*subinterface* {**l2transport** | **point-to-point**}] **Example:** `RP/0/RSP0/CPU0:router(config)# interface Multilink 0/1/0/0/100`	Creates a multilink bundle interface where you can specify Frame Relay encapsulation for the bundle. You create multilink Frame Relay subinterfaces under the multilink bundle interface.
Step 14	**encapsulation frame-relay** **Example:** `Router(config-if)# encapsulation frame-relay`	Specifies the Frame Relay encapsulation type.
Step 15	**frame-relay multilink bid** *bundle-id-name* **Example:** `Router(config-if)# frame-relay multilink bid MFRBundle`	**Note** (Optional) By default, the interface name, for example, Multilink 0/4/1/0/1, is used as the bundle identifier name. However, you can optionally create a name using the **frame-relay multilink bid** command.

Step 16	**frame-relay multilink bandwidth-class** {**a** | **b** | **c** [*threshold*]}
	Example:
	`Router(config-if)# frame-relay multilink bandwidth-class a`
		Configures one of three types of bandwidth classes on a multilink Frame Relay interface:
		• a—Bandwidth Class A
		• b—Bandwidth Class B
		• c—Bandwidth Class C
		The default is **a** (Bandwidth Class A).

	Command or Action	Purpose

Step 17	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits interface configuration mode.

Step 18	**`interface multilink`** *`interface-path-id`*`[.`*`subinterface`*
	`{`**`l2transport`**` | `**`point-to-point`**`}]`
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface Multilink 0/1/0/0/100.16 point-to-point`
		Creates a multilink subinterface in the *rack/slot/bay/controller-id bundleId.subinterace* [**point-to-point** | **l2transport**] notation and enters the subinterface configuration mode.
		• **l2transport**—Treat as an attachment circuit
		• **point-to-point**—Treat as a point-to-point link
		You can configure up to 992 subinterfaces on a multilink bundle interface. The DLCIs are 16 to 1007.

Step 19	**`ipv4 address`** *`ip-address`*
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 3.1.100.16 255.255.255.0`
		Assigns an IP address and subnet mask to the interface in the format:
		`A.B.C.D/prefix` `or` *`A.B.C.D/mask`*

Step 20	**`pvc`** *`dlci`*
	Example:
	`RP/0/RSP0/CPU0:router (config-subif)# pvc 16`
		Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.
		Replace *dlci* with a PVC identifier, in the range from 16 to 1007.
		Note Only one PVC is allowed per subinterface.

Step 21	**`service-policy`** `{`**`input`**` | `**`output`**`}` *`policy-map`*
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# service-policy output policy-mapA`
		Attaches a policy map to an input subinterface or output subinterface. When attached, the policy map is used as the service policy for the subinterface.
		Note For information on creating and configuring policy maps, refer to *Cisco IOS XR Modular Quality of Service Configuration Guide*.

Step 22	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# exit`
		Exits the Frame-Relay virtual circuit mode.

Step 23	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# exit`
		Exits the subinterface configuration mode.

Step 24	**`interface serial`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0`
		Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

	Command or Action	Purpose

Step 25	**`encapsulation mfr`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# encapsulation mfr`
		Enables multilink Frame Relay on the serial interface.

Step 26	**`multilink group`** *`group-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# multilink group 100`
		Specifies the multilink group ID for this interface.

Step 27	**`frame-relay multilink lid`** *`link-id name`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink lid sj1`
		Note Configures a name for the Frame Relay multilink bundle link.

Step 28	**`frame-relay multilink ack`** *`ack-timeout`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink ack 5`
		Configures the acknowledge timeout value for the Frame Relay multilink bundle link.

Step 29	**`frame-relay multilink hello`** *`hello-interval`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink hello 60`
		Configures the hello interval for the Frame Relay multilink bundle link.

Step 30	**`frame-relay multilink retry`** *`retry-count`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink retry 2`
		Configures the retry count for retransmissions for the Frame Relay multilink bundle link.

Step 31	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits interface configuration mode.

	Command or Action	Purpose

Step 32	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before`
		`exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 33	**`exit`**
	
	Example:
	`RP/0/RSP0/CPU0:router(config)# exit`
		Exits global configuration mode.

Step 34	**`show frame-relay multilink interface`** *`type interface-path-id`* `[`**`detail`** `|` **`verbose`**`]`
	
	Example:
	`RP/0/RSP0/CPU0:router# show frame-relay multilink interface Multilink 0/5/1/0/1`
		Shows the information retrieved from the interface description block (IDB), including bundle-specific information and Frame Relay information.

Configuring FRF.12 End-to-End Fragmentation on a Channelized Frame Relay Serial Interface

Perform the following steps to configure FRF.12 end-to-end fragmentation on a channelized Frame Relay serial interface.

SUMMARY STEPS

1. **config**
2. **controller t3** *interface-path-id*
3. **mode** *type*
4. **clock source** {**internal** | **line**}

5. **exit**
6. **controller t1** *interface-path-id*
7. **channel-group** *channel-group-number*
8. **timeslots** *range*
9. **exit**
10. **exit**
11. **interface serial** *interface-path-id*
12. **encapsulation frame-relay**
13. **exit**
14. **interface serial** *interface-path-id*
15. **ipv4 address** *ip-address*
16. **pvc** *dlci*
17. **service-policy** {**input** | **output**} *policy-map*
18. **fragment end-to-end** *fragment-size*
19. **exit**
20. **exit**
21. **exit**
22. **end**
    or
    commit
23. **exit**
24. **show frame-relay pvc** [**dlci** | **interface** | **location**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**config** **Example:** `RP/0/RSP0/CPU0:router# config`	Enters global configuration mode.
Step 2	**controller t3** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`	Specifies the T3 controller name in the *rack/slot/module/port* notation and enters T3 configuration mode.
Step 3	**mode** *type* **Example:** `RP/0/RSP0/CPU0:router(config-t3)# mode t1`	Configures the type of multilinks to channelize; for example, 28 T1s.

	Command or Action	Purpose

Step 4	**clock source** {**internal** | **line**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# clock source internal`
		(Optional) Sets the clocking for individual E3 links.
		Note The default clock source is **internal**.
		Note When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t3)# exit`	Exits T3/E3 or T1/E1 controller configuration mode.
Step 6	**controller t1** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# controller t1 0/1/0/0/0`	Enters T1 configuration mode.
Step 7	**channel-group** *channel-group-number* **Example:** `RP/0/RSP0/CPU0:router(config-t1)# channel-group 0`	Creates a T1 channel group and enters channel group configuration mode for that channel group.
Step 8	**timeslots** *range* **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24`	Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group. • Range is from 1 to 24 time slots. • You can assign all 24 time slots to a single channel group, or you can divide the time slots among several channel groups. **Note** Each individual T1 controller supports a total of 24 DS0 time slots.
Step 9	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit`	Exits channel group configuration mode.
Step 10	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-t1)# exit`	Exits T1 configuration mode.
Step 11	**interface serial** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0`	Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation.

	Command or Action	Purpose

Step 12	**`encapsulation frame-relay`**
	Example:
	`RP/0/RSP0/CPU0:Router(config-if)# encapsulation frame-relay`
		Specifies the Frame Relay encapsulation type.

Step 13	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits interface configuration mode.

Step 14	**`interface serial`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 1/0/0/0/0:0.1`
		Specifies the complete subinterface number with the *rack*/*slot*/*module*/*port*[/*channel-num:channel-group-number*].*subinterface* notation.

Step 15	**`ipv4 address`** *`ip-address`*
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 3.1.100.16 255.255.255.0`
		Assigns an IP address and subnet mask to the interface in the format:
		`A.B.C.D/prefix` `or` *`A.B.C.D/mask`*

Step 16	**`pvc`** *`dlci`*
	Example:
	`RP/0/RSP0/CPU0:router (config-subif)# pvc 100`
		Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.
		Replace *dlci* with a PVC identifier, in the range from 16 to 1007.
		Note Only one PVC is allowed per subinterface.

Step 17	**`service-policy`** `{`**`input`** `|` **`output`**`}` *`policy-map`*
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# service-policy output policy-mapA`
		Attaches a policy map to an input subinterface or output subinterface. When attached, the policy map is used as the service policy for the subinterface.
		Note For effective FRF.12 functionality (interleave specifically), you should configure an egress service policy with priority.
		Note For information on creating and configuring policy maps, refer to *Cisco IOS XR Modular Quality of Service Configuration Guide*,

Step 18	**`fragment end-to-end`** *`fragment-size`*
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# fragment end-to-end 100`
		(Optional) Enables fragmentation of Frame Relay frames on an interface and specifies the size (in bytes) of the payload from the original frame that will go into each fragment. This number excludes the Frame Relay header of the original frame.
		Valid values are from 64 to 512, depending on your hardware.

Step 19	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-fr-vc)# exit`
		Exits the Frame-Relay virtual circuit mode.

	Command or Action	Purpose

Step 20	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# exit`
		Exits the subinterface configuration mode.

Step 21	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits interface configuration mode.

Step 22	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 23	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# exit`
		Exits global configuration mode.

Step 24	**`show frame-relay pvc`** `[` **`dlci`** `|` **`interface`** `|` **`location`** `]`
	Example:
	`RP/0/RSP0/CPU0:router# show frame-relay pvc 100`
		Displays the information for the specified PVC DLCI, interface, or location.

Configuration Examples for Frame Relay

This section provides the following configuration examples:

- Optional Frame Relay Parameters: Example, page 573

- Multilink Frame Relay: Example, page 575
- End-to-End Fragmentation: Example, page 576

Optional Frame Relay Parameters: Example

The following example shows how to bring up and configure a POS interface with Frame Relay encapsulation. In this example, the user modifies the default Frame Relay configuration so that the interface supports ANSI T1.617a-1994 Annex D LMI on DCE.

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay IETF
RP/0/RSP0/CPU0:router(config-if)# frame-relay intf-type dce
RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi-type ansi
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# end

Uncommitted changes found, commit them? [yes]: yes

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router (config)# interface pos 0/3/0/0.10 point-to-point
RP/0/RSP0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24
RP/0/RSP0/CPU0:router (config-subif)# pvc 20
RP/0/RSP0/CPU0:router (config-fr-vc)# encap ietf
RP/0/RSP0/CPU0:router(config-subif)# commit
```

The following example shows how to disable LMI on a POS interface that has Frame Relay encapsulation configured:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface
RP/0/RSP0/CPU0:router(config)# interface pos 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi disable
RP/0/RSP0/CPU0:router(config-if)# end

Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to reenable LMI on a serial interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface
RP/0/RSP0/CPU0:router(config)# interface serial 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# no frame-relay lmi disable
RP/0/RSP0/CPU0:router(config-if)# end

Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to display Frame Relay statistics for LMI on all interfaces:

```
RP/0/RSP0/CPU0:router# show frame-relay lmi

LMI Statistics for interface POS0/1/0/0/ (Frame Relay DCE) LMI TYPE = ANSI
Invalid Unnumbered Info 0           Invalid Prot Disc 0
Invalid Dummy Call Ref 0            Invalid Msg Type 0
Invalid Status Message 0            Invalid Lock Shift 9
Invalid Information ID 0            Invalid Report IE Len 0
Invalid Report Request 0            Invalid Keep IE Len 0
Num Status Enq. Rcvd 9444           Num Status Msgs Sent 9444
Num Full Status Sent 1578           Num St Enq. Timeouts 41
Num Link Timeouts 7
```

```
LMI Statistics for interface POS0/1/0/1/ (Frame Relay DCE) LMI TYPE = CISCO
Invalid Unnumbered Info 0            Invalid Prot Disc 0
Invalid Dummy Call Ref 0             Invalid Msg Type 0
Invalid Status Message 0             Invalid Lock Shift 0
Invalid Information ID 0             Invalid Report IE Len 0
Invalid Report Request 0             Invalid Keep IE Len 0
Num Status Enq. Rcvd 9481            Num Status Msgs Sent 9481
Num Full Status Sent 1588            Num St Enq. Timeouts 16
Num Link Timeouts 4
```

The following example shows how to create a serial subinterface with a PVC on the main serial interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface serial 0/3/0/0/0:0.10 point-to-point
RP/0/RSP0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24
RP/0/RSP0/CPU0:router (config-subif)# pvc 20
RP/0/RSP0/CPU0:router (config-fr-vc)# encapsulation ietf
RP/0/RSP0/CPU0:router(config-subif)# commit
```

The following example shows how to display information about all PVCs configured on your system:

```
RP/0/RSP0/CPU0router# show frame-relay pvc

PVC Statistics for interface Serial0/3/2/0 (Frame Relay DCE)

              Active     Inactive      Deleted       Static
  Local          4           0            0             0
  Switched       0           0            0             0
  Dynamic        0           0            0             0

DLCI = 612, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.1
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00

DLCI = 613, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.2
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00

DLCI = 614, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.3
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00

DLCI = 615, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.4
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00
```

The following example shows how to modify LMI polling options on PVCs configured for a DTE, and then use the **show frame-relay lmi** and **show frame-relay lmi-info** commands to display information for monitoring and troublehooting the interface:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface pos 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi-n391dte 10
RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi-n391dte 5
RP/0/RSP0/CPU0:router(config-if)# frame-relay lmi-t391dte 15
RP/0/RSP0/CPU0:router(config-subif)# commit

RP/0/RSP0/CPU0:router# show frame-relay lmi interface pos 0/3/0/0

LMI Statistics for interface pos 0/3/0/0 (Frame Relay DTE) LMI TYPE = ANSI
Invalid Unnumbered Info 0            Invalid Prot Disc 0
Invalid Dummy Call Ref 0             Invalid Msg Type 0
Invalid Status Message 0             Invalid Lock Shift 9
Invalid Information ID 0             Invalid Report IE Len 0
Invalid Report Request 0             Invalid Keep IE Len 0
Num Status Enq. Rcvd 9444            Num Status Msgs Sent 9444
Num Full Status Sent 1578            Num St Enq. Timeouts 41
Num Link Timeouts 7

RP/0/RSP0/CPU0:router# show frame-relay lmi-info interface pos 0/3/0/0

LMI IDB Info for interface POS0/3/0/0
  ifhandle:             0x6176840
  Interface type:       DTE
  Interface state:      UP
  Line Protocol:        UP
  LMI type (cnf/oper):  AUTO/CISCO
  LMI type autosense:   OFF
  Interface MTU:        1504
  -------------- DTE -------------
  T391:                 15s
  N391: (cnf/oper):     5/5
  N392: (cnf/oper):     3/0
  N393:                 4
  My seq#:              83
  My seq# seen:         83
  Your seq# seen:       82
  -------------- DCE -------------
  T392:                 15s
  N392: (cnf/oper):     3/0
  N393:                 4
  My seq#:              0
  My seq# seen:         0
  Your seq# seen:       0
```

Multilink Frame Relay: Example

The following example shows how to configure multilink Frame Relay with serial interfaces:

```
RP/0/RSP0/CPU0:router# config
RP/0/RSP0/CPU0:router(config)# controller MgmtMultilink 0/3/1/0
RP/0/RSP0/CPU0:router(config-mgmtmultilink)# bundle 100
RP/0/RSP0/CPU0:router(config-mgmtmultilink)# exit

RP/0/RSP0/CPU0:router(config)# controller T3 0/3/1/0
RP/0/RSP0/CPU0:router(config-t3)# mode t1
```

```
RP/0/RSP0/CPU0:router(config-t3)# clock source internal
RP/0/RSP0/CPU0:router(config-t3)# exit

RP/0/RSP0/CPU0:router(config)# controller T1 0/3/1/0/0
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)#  exit
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# exit

RP/0/RSP0/CPU0:router(config)# interface Multilink 0/3/1/0/100
RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/RSP0/CPU0:router(config-if)# exit

RP/0/RSP0/CPU0:router(config)# interface Multilink 0/3/1/0/100.16 point-to-point
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 3.1.100.16 255.255.255.0
RP/0/RSP0/CPU0:router(config-subif)# pvc 16
RP/0/RSP0/CPU0:router(config-fr-vc)# service-policy output policy-mapA
RP/0/RSP0/CPU0:router(config-fr-vc)# exit
RP/0/RSP0/CPU0:router(config-subif)# exit

RP/0/RSP0/CPU0:router(config)# interface Serial 0/3/1/0/0:0
RP/0/RSP0/CPU0:router(config-if)# encapsulation mfr
RP/0/RSP0/CPU0:router(config-if)# multilink group 100
RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink lid sj1
RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink ack 5
RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink hello 60
RP/0/RSP0/CPU0:router(config-if)# frame-relay multilink retry 2
RP/0/RSP0/CPU0:router(config-if)# exit
RP/0/RSP0/CPU0:router(config)#
```

End-to-End Fragmentation: Example

The following example shows how to configure FRF.12 end-to-end fragmentation on a channelized Frame Relay serial interface:

```
RP/0/RSP0/CPU0:router# config
RP/0/RSP0/CPU0:router(config)# controller T30/3/1/0
RP/0/RSP0/CPU0:router(config-t3)# mode t1
RP/0/RSP0/CPU0:router(config-t3)# clock source internal
RP/0/RSP0/CPU0:router(config-t3)# exit
RP/0/RSP0/CPU0:router(config-t3)# controller T10/3/1/0/0
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)#  timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1-channel_group)# interface Serial 0/3/1/0/0:0
RP/0/RSP0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/RSP0/CPU0:router(config-if)# exit
RP/0/RSP0/CPU0:router(config-if)# interface Serial 0/3/1/0/0:0.100 point-to-point
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 3.1.1.1 255.255.255.0
RP/0/RSP0/CPU0:router(config-subif)#  pvc 100
RP/0/RSP0/CPU0:router(config-fr-vc)#  service-policy output LFI
RP/0/RSP0/CPU0:router(config-fr-vc)# fragment end-to-end 256
```

Additional References

The following sections provide references related to Frame Relay.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using Cisco IOS XR software	*Cisco IOS XR Getting Started Guide*
Cisco IOS XR AAA services configuration information	*Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference*

Standards

Standards	Title
FRF.12	*Frame Relay Forum .12*
FRF.16	*Frame Relay Forum .16*
ANSI T1.617 Annex D	*American National Standards Institute T1.617 Annex D*
ITU Q.933 Annex A	*International Telecommunication Union Q.933 Annex A*

MIBs

MIBs	MIBs Link
FRF.16 MIB Cisco Frame Relay MIB IF-MIB Management Information Base for Frame Relay DTEs Management Information Base for Frame Relay DTEs Using SMIv2	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1294	*Multiprotocol Interconnect Over Frame Relay*
RFC 1315	*Management Information Base for Frame Relay DTEs*
RFC 1490	*Multiprotocol Interconnect Over Frame Relay*
RFC 1586	*Guidelines for Running OSPF Over Frame Relay Networks*
RFC 1604	*Definitions of Managed Objects for Frame Relay Service*
RFC 2115	*Management Information Base for Frame Relay DTEs Using SMIv2*
RFC 2390	*Inverse Address Resolution Protocol*

RFCs	Title
RFC 2427	*Multiprotocol Interconnect Over Frame Relay*
RFC 2954	*Definitions of Managed Objects for Frame Relay Service*
RFC 3020	*RFC for FRF.16 MIB*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring PPP on the Cisco ASR 9000 Series Router

This module describes the configuration of Point-to-Point Protocol (PPP) on POS and serial interfaces on the Cisco ASR 9000 Series Router.

Feature History for Configuring PPP Interfaces

Release	Modification
Release 3.9.0	PPP and ICSSO for PPP and MLPPP were introduced on the Cisco ASR 9000 Series Router.
Release 3.9.1	Support for T3 Channelized SONET was added.
Release 4.0.0	Support for the following features was added for the 2-Port Channelized OC-12c/DS0 SPA: • IPHC over PPP, MLPPP, and MLPPP/LFI • NxDS0 serial interfaces Support for PPP was introduced on the following SPAs: • 1-Port Channelized OC-48/STM-16 SPA • 1-Port OC-192c/STM-64 POS/RPR XFP SPA • 2-Port OC-48c/STM-16 POS/RPR SPA • 8-Port OC-12c/STM-4 POS SPA

Release 4.0.1	Support for PPP was added for the following SPAs on the Cisco ASR 9000 Series Router: • Cisco 1-Port Channelized OC-3/STM-1 SPA (also supports MLPPP) • Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA • Cisco 4-Port OC-3c/STM-1 SPA • Cisco 8-Port OC-3c/STM-1 SPA
Release 4.1.0	Support for the Noise Attribute was added for PPP to remove links on MLPPP bundles when Link Noise Monitoring (LNM) thresholds are crossed on a link. Support for PPP, including MLPPP support on T1/E1 channels, was introduced on the following SPAs: • Cisco 4-Port Channelized T3 SPA • Cisco 8-Port Channelized T1/E1 SPA

Contents

Prerequisites for Configuring PPP

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before you can configure PPP authentication on a POS or serial interface, be sure that the following tasks and conditions are met:

- Your hardware must support POS or serial interfaces.
- You have enabled PPP encapsulation on your interface with the **encap ppp** command, as described in the appropriate module:
  - To enable PPP encapsulation on a POS interface, see the *Configuring POS Interfaces onthe Cisco ASR 9000 Series Router* module in this manual.
  - To enable PPP encapsulation on a serial interface, see the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module in this manual.

Information About PPP

To configure PPP and related features, you should understand the information in this section:

PPP Authentication

When PPP authentication is configured on an interface, a host requires that the other host uniquely identify itself with a secure password before establishing a PPP connection. The password is unique and is known to both hosts.

PPP supports the following authentication protocols:

- Challenge-Handshake Authentication Protocol (CHAP)
- Microsoft extension to the CHAP protocol (MS-CHAP)
- Password Authentication Protocol (PAP).

When you first enable PPP on a POS or serial interface, no authentication is enabled on the interface until you configure a CHAP, MS-CHAP, or PAP secret password under that interface. Keep the following information in mind when configuring PPP on an interface:

- CHAP, MS-CHAP, and PAP can be configured on a single interface; however, only one authentication method is used at any one time. The order in which the authentication protocols are used is determined by the peer during the LCP negotiations. The first authentication method used is the one that is also supported by the peer.
- PAP is the least secure authentication protocol available on POS and serial interfaces. To ensure higher security for information that is sent over POS and serial interfaces, we recommend configuring CHAP or MS-CHAP authentication in addition to PAP authentication.
- Enabling or disabling PPP authentication does not effect the local router’s willingness to authenticate itself to the remote device.
- The **ppp authentication** command is also used to specify the order in which CHAP, MS-CHAP, and PAP authentication is selected on the interface. You can enable CHAP, MS-CHAP, or PAP in any order. If you enable all three methods, the first method specified is requested during link negotiation. If the peer suggests using the second method, or refuses the first method, the second method is tried. Some remote devices support only one method. Base the order in which you specify methods on the remote device’s ability to correctly negotiate the appropriate method and on the level of data line security you require. PAP usernames and passwords are sent as clear text strings, which can be intercepted and reused.

Caution If you use a *list-name* value that was not configured with the **aaa authentication ppp** command, your interface cannot authenticate the peer. For details on implementing the **aaa authentication** command with the **ppp** keyword, see the *Authentication, Authorization, and Accounting Commands on Cisco IOS XR Software* module of *Cisco IOS XR System Security Command Reference* and *Configuring AAA Services on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Configuration Guide*.

PAP Authentication

PAP provides a simple method for a remote node to establish its identity using a two-way handshake. After a PPP link is established between two hosts, a username and password pair is repeatedly sent by the remote node across the link (in clear text) until authentication is acknowledged, or until the connection is terminated.

PAP is not a secure authentication protocol. Passwords are sent across the link in clear text and there is no protection from playback or trial-and-error attacks. The remote node is in control of the frequency and timing of the login attempts.

CHAP Authentication

CHAP is defined in RFC 1994, and it verifies the identity of the peer by means of a three-way handshake. The steps that follow provide a general overview of the CHAP process:

Step 1 The CHAP authenticator sends a challenge message to the peer.

Step 2 The peer responds with a value calculated through a one-way hash function.

Step 3 The authenticator checks the response against its own calculation of the expected hash value. If the values match, then the authentication is successful. If the values do not match, then the connection is terminated.

This authentication method depends on a CHAP password known only to the authenticator and the peer. The CHAP password is not sent over the link. Although the authentication is only one-way, you can negotiate CHAP in both directions, with the help of the same CHAP password set for mutual authentication.

Note For CHAP authentication to be valid, the CHAP password must be identical on both hosts.

MS-CHAP Authentication

Microsoft Challenge Handshake Authentication Protocol (MS-CHAP) is the Microsoft version of CHAP and is an extension to RFC 1994. MS-CHAP follows the same authentication process used by CHAP. In this case, however, authentication occurs between a PC using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server (NAS).

Note For MS-CHAP authentication to be valid, the MS-CHAP password must be identical on both hosts.

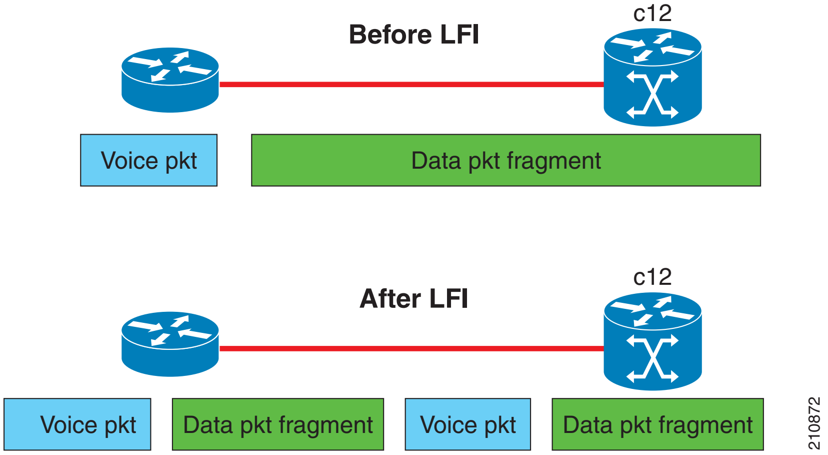
Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) provides a method for combining multiple physical links into one logical link. The implementation combines multiple PPP interfaces into one multilink interface. MLPPP performs the fragmenting, reassembling, and sequencing of datagrams across multiple PPP links.

Link Fragmentation and Interleaving (LFI) is designed for MLPPP interfaces and is required when integrating voice and data on low-speed interfaces.

Link Fragmentation and Interleaving (LFI) provides stability for delay-sensitive traffic, such as voice or video, traveling on the same circuit as data. Voice is susceptible to increased latency and jitter when the network processes large packets on low-speed interfaces. LFI reduces delay and jitter by fragmenting large datagrams and interleaving them with low-delay traffic packets.

Figure 34 Link Fragmentation Interleave



MLPPP Feature Summary

MLPPP in Cisco IOS XR provides the same features that are supported on PPP Serial interfaces with the exception of QoS. It also provides the following additional features:

- Long sequence numbers (24-bit).
- Lost fragment detection timeout period of 1 second.
- Minimum-active-links configuration option.
- LCP echo request/reply support over multilink interface.
- Full T1 and E1 framed and unframed links.
- Support for the Cisco 2-Port Channelized OC-12c/DS0 SPA to set thresholds for noise errors on T1/E1 links that are used to signal the Noise Attribute to PPP for removal of an MLPPP bundle link. For more information about LNM, see the “Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on the Cisco ASR 9000 Series Router” module in the *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide*.

IPHC Over MLPPP

The 2-Port Channelized OC-12c/DS0 SPA supports IPHC over PPP, MLPPP, and MLPPP/LFI. For more information about IPHC and how to configure it, see the “Configuring Serial Interfaces on the Cisco ASR 9000 Series Router” module in the *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide*.

ICSSO for PPP and MLPPP

Note SR- and MR-APS is not supported on the Cisco 1-Port Channelized OC-48/STM-16 SPA.

Inter-Chassis Stateful Switchover (ICSSO) on the Cisco ASR 9000 Series Router provides features that maintain Point-to-Point Protocol (PPP) and Multilink PPP (MLPPP) sessions during a Multi-Router Automatic Protection Switching (MR-APS) switchover from the MR-APS Working router to the MR-APS Protect router.

ICSSO allows an MR-APS switchover to occur without the need for Link Control Protocol (LCP) or IP Control Protocol (IPCP) renegotiation between the new MR-APS active router and the remote PPP/MLPPP peer devices. The primary purpose of ICSSO is to minimize subscriber session and data loss during an MR-APS switchover.

ICSSO synchronizes the PPP and MLPPP state information on the active router with the state information on the backup router, and ensures that the backup router is ready to forward traffic immediately after an MR-APS switchover.

ICSSO works in conjunction with the following other software components:

- Multi-Router Automatic Protection Switching (MR-APS), page 584
- Session State Redundancy Protocol (SSRP), page 584
- Redundancy Group Manager (RG-MGR), page 585
- IP Fast Reroute (IP-FRR), page 585
- VPN Routing And Forwarding (VRF), page 585
- Open Shortest Path First (OSPF), page 586

Multi-Router Automatic Protection Switching (MR-APS)

Multi-Router Automatic Protection Switching (MR-APS) is a Cisco feature that provides Layer 1 protection against facility and equipment failures through the configuration of a protection pair of SONET controllers located on two different routers. The redundant backup router is configured identically to the active router and is ready to forward traffic immediately upon an MR-APS switchover.

The protection pair communicates using Layer 1 (k1/k2) signalling bytes from the SONET downstream connection (as per Bellcore specification GR-253-CORE) and Layer 3 signaling messages using Protect Group Protocol (PGP). MR-APS detects many of the sources of failures that indirectly trigger an IP-FRR update to use backup routes.

In an MR-APS configuration, two interfaces, on different routers, are assigned the roles of Working interface or Protect interface. These roles are configured by the operator. Under normal conditions, the Working interface carries active traffic. If the Working interface fails, the Protect interface takes over the active traffic immediately with no loss of PPP traffic.

Session State Redundancy Protocol (SSRP)

A pair of SONET controllers configured for MR-APS are part of a Session State Redundancy Protocol (SSRP) protection group. SSRP communicates interface and system state information between the Active and Standby routers. SSRP also serves as the keepalive protocol.

SSRP configuration associates a SONET controller with an inter-chassis redundancy group and enables MR-APS peer routers to synchronize PPP session states on each Active SONET controller.

PPP sessions can have one of three states:

- Active–A PPP session is in the Active state when the PPP session negotiation is complete, the associated route is installed, and the associated adjacency is created. PPP sessions in the Active state replicate data to their peers on the Standby router.
- Standby Up–A PPP session on the Standby router is in the Standby Up state when replicated state information is received from the Active router, the associated PPP route is installed, and the associated adjacency is created. PPP sessions in the Standby Up state are ready to forward traffic immediately after an MR-APS switchover.
- Standby Down–A PPP session on the Standby router is in the Standby Down state when the associated route is not installed and the adjacency is not created.

SSRP runs between the MR-APS peer routers and uses TCP/IP. One SSRP session runs on each pair of redundant SONET controllers, meaning multiple SSRP sessions can be running on a pair of MR-APS-redundant routers.

Note SSRP is not a redundancy control protocol, but is a state information synchronization protocol.

Redundancy Group Manager (RG-MGR)

The Redundancy Group Manager (RG-MGR) configures the backup routes for the protected interface. The RG-MGR registers events on protected SONET controllers and provides the Routing Information Base (RIB) component with IP Fast Reroute (IP-FRR) updates.

IP Fast Reroute (IP-FRR)

Note IP-FRR, when used with IC-SSO, is only supported with PPP encapsulation. It is not supported with HDLC encapsulation.

IP Fast Reroute (IP-FRR) provides extremely fast rerouting of PPP/MLPPP traffic after an MR-APS switchover.

IP-FRR controls the primary and backup routes. Each route is mapped in the Routing Information Base (RIB), and IP-FRR controls which backup path is used to forward traffic after an MR-APS switchover.

An MR-APS switchover triggers an IP-FRR update, which activates the backup routes on the protection SONET controller. When the working SONET controller is restored, another IP-FRR update is triggered, and traffic is rerouted to the primary route.

For more information about IP-FRR, refer to the “Implementing MPLS Traffic Engineering on Cisco IOS XR Software” module in the *Cisco IOS XR MPLS Configuration Guide*.

VPN Routing And Forwarding (VRF)

ICSSO can be used with VPN routing and forwarding (VRF). Customers who wish to isolate traffic streams with different service types can do so using VRF technology. VRF allows the user to create and maintain separate routing and forwarding databases. See VRF on Multilink Configuration for Use with ICSSO: Example, page 625 and VRF on Ethernet Configuration for Use with ICSSO: Example, page 625. For more information on configuring VRF, refer to the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*.

Open Shortest Path First (OSPF)

Aggregation routers that terminate PPP sessions to a set of remote peers, must advertise their availability on the network using Open Shortest Path First (OSPF). OSPF is required to advertise the availability of remote PPP peers to the ICSSO peer router. See OSPF Configuration for Use with ICSSO: Example, page 626. For more information on configuring OSPF, refer to the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*.

ICSSO Configuration Overview

ICSSO is configured as follows:

- Configure MR-APS
- Configure SSRP profile
- Configure SSRP groups
- Configure serial interfaces with PPP encapsulation
- Configure multilink interfaces
- Verify ICSSO configuration

The "Configuring ICSSO for PPP and MLPPP" section on page 612 of this module provides step procedures for configuring ICSSO.

The "ICSSO for PPP and MLPPP Configuration: Examples" section on page 622 gives specific examples for configuring ICSSO and related components.

Multiclass MLPPP with QoS

Multiclass Multilink Point-to-Point Protocol (MLPPP) can be utilized with Quality of Service (QoS) and configured using the **encap-sequence** command under a class in a policy map.

The **encap-sequence** command specifies the MLPPP MCMP class ID for the packets in an MQC defined class.

The valid values for the **encap-sequence** ID number are **none**, 0, 1, 2, or 3. The **none** value is applicable only when the **priority level** is 1 and indicates that there is no MLPPP encapsulation. The values 1, 2, or 3 can be used with priority 1 or 2 classes or other classes with queuing actions. An **encap-sequence** ID number of zero (0) is reserved for the default class and cannot be specified in any other classes.

Note The **encap-sequence** ID numbers must be configured in numeric order. For example, you cannot assign an ID number of 3 unless you have already assigned 1 and 2.

The number of **encap-sequence** ID numbers must be less than the number of MLPPP classes that are negotiated between the peers via the Multilink header. The user must ensure that the configuration is consistent as the system does not verify this.

The **ppp multilink multiclass remote apply** command provides a way to ensure this. You can ensure that the number of classes using an **encap-sequence** ID number (including the default of 0) is less than the *min-number* value in the **ppp multilink multiclass remote apply** command. For example, if the min-number value in the ppp multilink multiclass remote apply command is 4, you can only have 3 or less classes with encap-sequence ID numbers

The QoS policy validates the following conditions. If these conditions are not met, the policy is rejected:

- The **encap-sequence** ID number is within the allowed values of 1 to 3.
- When **encap-sequence** is configured for any class in a policy map, all classes in that policy map with **priority level 1** must also contain an **encap-sequence** ID number.
- The **encap-sequence none** configuration is restricted to classes with **priority level** 1.
- The class-default does not contain an **encap-sequence** configuration.
- Only classes containing a queuing action have the **encap-sequence** configuration.

Note Classes that share the same **encap-sequence** ID number must have the same priority.

A QoS policy map is configured as follows:

```
config
   policy-map type qos policy-name
      class class-name
         action
         action
         action
. . .
```

The following example shows how to configure a policy map for MLPPP:

```
config
   policy-map foo
      class ip-prec-1
         encap-sequence none
         police rate percent 10
         priority level 1
   !
      class ip-prec-2
         encap-sequence 1
         shape average percent 80
   !
      class ip-prec-3
         encap-sequence 1
         bandwidth percent 10
   !
      class class-default
   !
   end-policy-map
   !
```

For complete information on configuring QoS and QoS commands, refer to the *Cisco ASR 9000 Series Aggregation Services Routers Modular Quality of Service Configuration Guide* and the *Cisco ASR 9000 Series Aggregation Services Routers Modular Quality of Service Command Reference*.

T3 SONET Channels

The Cisco ASR 9000 Series Router supports T3 channelized SONET on the following hardware:

- SIP 700 SPA Interface Processor
- 1-Port Channelized OC-3/STM-1 SPA
- 2-Port Channelized OC-12c/DS0 SPA
- 1-Port Channelized OC-48/STM-16 SPA
-

Channelized SONET provides the ability to transport multiple T3 channels over the same physical link.

For more detailed information about configuring channelized SONET, T3 and T1 controllers, serial interfaces, and SONET APS, see the following related modules:

- "Configuring Channelized SONET/SDH on the Cisco ASR 9000 Series Router"
- "Configuring Clear Channel SONET Controllers on the Cisco ASR 9000 Series Router"
- "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on the Cisco ASR 9000 Series Router"
- "Configuring Serial Interfaces on the Cisco ASR 9000 Series Router"

How to Configure PPP

This section includes the following procedures:

- Modifying the Default PPP Configuration, page 588
- Configuring PPP Authentication, page 591
- Disabling an Authentication Protocol, page 599
- Configuring Multilink PPP, page 604
- Configuring ICSSO for PPP and MLPPP, page 612

Modifying the Default PPP Configuration

When you first enable PPP on an interface, the following default configuration applies:

- The interface resets itself immediately after an authentication failure.
- The maximum number of configuration requests without response permitted before all requests are stopped is 10.
- The maximum number of consecutive Configure Negative Acknowledgments (CONFNAKs) permitted before terminating a negotiation is 5.
- The maximum number of terminate requests (TermReqs) without response permitted before the Link Control Protocol (LCP) or Network Control Protocol (NCP) is closed is 2.
- Maximum time to wait for a response to an authentication packet is 10 seconds.
- Maximum time to wait for a response during PPP negotiation is 3 seconds.

This task explains how to modify the basic PPP configuration on serial and POS interfaces that have PPP encapsulation enabled. The commands in this task apply to all authentication types supported by PPP (CHAP, MS-CHAP, and PAP).

Prerequisites

You must enable PPP encapsulation on the interface with the **encapsulation ppp** command.

- To enable PPP encapsulation on a POS interface, see the *Configuring POS Interfaces onthe Cisco ASR 9000 Series Router* module in this manual.
- To enable PPP encapsulation on an interface, see the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module in this manual.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp max-bad-auth** *retries*
4. **ppp max-configure** *retries*
5. **ppp max-failure** *retries*
6. **ppp max-terminate** *number*
7. **ppp timeout authentication** *seconds*
8. **ppp timeout retry** *seconds*
9. **end**
   or
   commit
10. **show ppp interfaces** {*type interface-path-id* | **all** | **brief** {*type interface-path-id* | **all** | **location** *node-id*} | **detail** {*type interface-path-id* | **all** | **location** *node-id*} | **location** *node-id*}

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`ppp max-bad-auth`** *`retries`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ppp max-bad-auth 3`
		(Optional) Configures the number of authentication retries allowed on an interface after a PPP authentication failure.
		• If you do not specify the number of authentication retries allowed, the router resets itself immediately after an authentication failure.
		• Replace the *retries* argument with number of retries after which the interface is to reset itself, in the range from 0 through 10.
		• The default is 0 retries.
		• The **ppp max-bad-auth** command applies to any interface on which PPP encapsulation is enabled.

	Command or Action	Purpose
Step 4	`ppp max-configure` *retries* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp max-configure 4`	(Optional) Specifies the maximum number of configure requests to attempt (without response) before the requests are stopped. • Replace the *retries* argument with the maximum number of configure requests retries, in the range from 4 through 20. • The default maximum number of configure requests is 10. • If a configure request message receives a reply before the maximum number of configure requests are sent, further configure requests are abandoned.
Step 5	`ppp max-failure` *retries* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp max-failure 3`	(Optional) Configures the maximum number of consecutive Configure Negative Acknowledgments (CONFNAKs) permitted before a negotiation is terminated. • Replace the *retries* argument with the maximum number of CONFNAKs to permit before terminating a negotiation, in the range from 2 through 10. • The default maximum number of CONFNAKs is 5.
Step 6	`ppp max-terminate` *number* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp max-terminate 5`	(Optional) Configures the maximum number of terminate requests (TermReqs) to send without reply before the Link Control Protocol (LCP) or Network Control Protocol (NCP) is closed. • Replace the *number* argument with the maximum number of TermReqs to send without reply before closing down the LCP or NCP. Range is from 2 to 10. • The default maximum number of TermReqs is 2.
Step 7	`ppp timeout authentication` *seconds* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp timeout authentication 20`	(Optional) Sets PPP authentication timeout parameters. • Replace the *seconds* argument with the maximum time, in seconds, to wait for a response to an authentication packet. Range is from 3 to 30 seconds. • The default authentication time is 10 seconds, which should allow time for a remote router to authenticate and authorize the connection and provide a response. However, it is also possible that it will take much less time than 10 seconds. In such cases, use the **ppp timeout authentication** command to lower the timeout period to improve connection times in the event that an authentication response is lost.
Step 8	`ppp timeout retry` *seconds* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp timeout retry 8`	(Optional) Sets PPP timeout retry parameters. • Replace the *seconds* argument with the maximum time, in seconds, to wait for a response during PPP negotiation. Range is from 1 to 10 seconds. • The default is 3 seconds.

	Command or Action	Purpose

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 10	**show ppp interfaces** {*type interface-path-id* | **all** | **brief** {*type interface-path-id* | **all** | **location** *node-id*} | **detail** {*type interface-path-id* | **all** | **location** *node-id*} | **location** *node-id*}
	Example:
	`RP/0/RSP0/CPU0:router# show ppp interfaces serial 0/2/0/0`
		Verifies the PPP configuration for an interface or for all interfaces that have PPP encapsulation enabled.

Configuring PPP Authentication

This section contains the following procedures:

- Enabling PAP, CHAP, and MS-CHAP Authentication, page 591
- Configuring a PAP Authentication Password, page 594
- Configuring a CHAP Authentication Password, page 596
- Configuring an MS-CHAP Authentication Password, page 598

Enabling PAP, CHAP, and MS-CHAP Authentication

This task explains how to enable PAP, CHAP, and MS-CHAP authentication on a serial or POS interface.

Prerequisites

You must enable PPP encapsulation on the interface with the **encapsulation ppp** command, as described in the following modules:

- To enable PPP encapsulation on a POS interface, see the *Configuring POS Interfaces onthe Cisco ASR 9000 Series Router* module in this manual.
- To enable PPP encapsulation on an interface, see the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module in this manual.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp authentication** *protocol* [*protocol* [*protocol*]] [*list-name* | **default**]
4. **end**
   or
   commit
5. **show ppp interfaces** {*type interface-path-id* | **all** | **brief** {*type interface-path-id* | **all** | **location** *node-id*} | **detail** {*type interface-path-id* | **all** | **location** *node-id*} | **location** *node-id*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters interface configuration mode.
Step 3	**ppp authentication** *protocol* [*protocol* [*protocol*]] [*list-name* | **default**] **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp authentication chap pap MIS-access`	Enables CHAP, MS-CHAP, or PAP on an interface, and specifies the order in which CHAP, MS-CHAP, and PAP authentication is selected on the interface. • Replace the *protocol* argument with **pap**, **chap**, **or ms-chap**. • Replace the *list name* argument with the name of a list of methods of authentication to use. To create a list, use the **aaa authentication ppp** command, as described in the *Authentication, Authorization, and Accounting Commands on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Command Reference*. • If no list name is specified, the system uses the default. The default list is designated with the **aaa authentication ppp** command, as described in the *Authentication, Authorization, and Accounting Commands on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Command Reference*.

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show ppp interfaces** {*type interface-path-id* | **all** | **brief** {*type interface-path-id* | **all** | **location** *node-id*} | **detail** {*type interface-path-id* | **all** | **location** *node-id*} | **location** *node-id*}
	
	Example:
	`RP/0/RSP0/CPU0:router# show ppp interfaces serial 0/2/0/0`
		Displays PPP state information for an interface.
		• Enter the *type interface-path-id* argument to display PPP information for a specific interface.
		• Enter the **brief** keyword to display brief output for all interfaces on the router, for a specific interface instance, or for all interfaces on a specific node.
		• Enter the **all** keyword to display detailed PPP information for all nodes installed in the router.
		• Enter the **location** *node-id* keyword argument to display detailed PPP information for the designated node.
		There are seven possible PPP states applicable for either the Link Control Protocol (LCP) or the Network Control Protocol (NCP).

Where To Go Next

Configure a PAP, CHAP, or MS-CHAP authentication password, as described in the appropriate section:

- If you enabled PAP on an interface, configure a PAP authentication username and password, as described in the "Configuring a PAP Authentication Password" section on page 594.
- If you enabled CHAP on an interface, configure a CHAP authentication password, as described in the "Configuring a CHAP Authentication Password" section on page 596
- If you enabled MS-CHAP on an interface, configure an MS-CHAP authentication password, as described in the "Configuring an MS-CHAP Authentication Password" section on page 598

Configuring a PAP Authentication Password

This task explains how to enable and configure PAP authentication on a serial or POS interface.

Note PAP is the least secure authentication protocol available on POS and interfaces. To ensure higher security for information that is sent over POS and interfaces, we recommend configuring CHAP or MS-CHAP authentication in addition to PAP authentication.

Prerequisites

You must enable PAP authentication on the interface with the **ppp authentication** command, as described in the "Enabling PAP, CHAP, and MS-CHAP Authentication" section on page 591.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp pap sent-username** *username* **password** [**clear** | **encrypted**] *password*
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`ppp pap sent-username`** *`username`* **`password`** `[`**`clear`** `|` **`encrypted`**`]` *`password`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ppp pap sent-username xxxx password notified`
		Enables remote Password Authentication Protocol (PAP) support for an interface, and includes the **sent-username** and **password** commands in the PAP authentication request packet to the peer.
		• Replace the *username* argument with the username sent in the PAP authentication request.
		• Enter **password clear** to select cleartext encryption for the password, or enter **password encrypted** if the password is already encrypted.
		• The **ppp pap sent-username** command allows you to replace several username and password configuration commands with a single copy of this command on interfaces.
		• You must configure the **ppp pap sent-username** command for each interface.
		• Remote PAP support is disabled by default.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show running-config`**
	Example:
	`RP/0/RSP0/CPU0:router# show running-config`
		Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

Configuring a CHAP Authentication Password

This task explains how to enable CHAP authentication and configure a CHAP password on a serial or POS interface.

Prerequisites

You must enable CHAP authentication on the interface with the **ppp authentication** command, as described in the "Enabling PAP, CHAP, and MS-CHAP Authentication" section on page 591.

Restrictions

The same CHAP password must be configured on both host endpoints.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp chap password** [**clear** | **encrypted**] *password*
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`ppp chap password`** `[`**`clear`** `|` **`encrypted`**`]` *`password`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ppp chap password clear xxxx`
		Enables CHAP authentication on the specified interface, and defines an interface-specific CHAP password.
		• Enter **clear** to select cleartext encryption, or **encrypted** if the password is already encrypted.
		• Replace the *password* argument with a cleartext or already-encrypted password. This password is used to authenticate secure communications among a collection of routers.
		• The **ppp chap password** command is used for remote CHAP authentication only (when routers authenticate to the peer) and does not effect local CHAP authentication.This command is useful when you are trying to authenticate a peer that does not support this command (such as a router running an older Cisco IOS XR software image).
		• The CHAP secret password is used by the routers in response to challenges from an unknown peer.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show running-config`**
	Example:
	`RP/0/RSP0/CPU0:router# show running-config`
		Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

Configuring an MS-CHAP Authentication Password

This task explains how to enable MS-CHAP authentication and configure an MS-CHAP password on a serial or POS interface.

Prerequisites

You must enable MS-CHAP authentication on the interface with the **ppp authentication** command, as described in the "Enabling PAP, CHAP, and MS-CHAP Authentication" section on page 591.

Restrictions

The same MS-CHAP password must be configured on both host endpoints.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp ms-chap password** [**clear** | **encrypted**] *password*
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters interface configuration mode.
Step 3	**ppp ms-chap password** [**clear** | **encrypted**] *password* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ppp ms-chap password clear xxxx`	Enables a router calling a collection of routers to configure a common Microsoft Challenge Handshake Authentication (MS-CHAP) secret password. The MS-CHAP secret password is used by the routers in response to challenges from an unknown peer.

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show running-config** **Example:** `RP/0/RSP0/CPU0:router# show running-config`	Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

Disabling an Authentication Protocol

This section contains the following procedures:

- Disabling PAP Authentication on an Interface, page 599
- Disabling CHAP Authentication on an Interface, page 601
- Disabling MS-CHAP Authentication on an Interface, page 602

Disabling PAP Authentication on an Interface

This task explains how to disable PAP authentication on a serial or POS interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp pap refuse**
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`ppp pap refuse`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ppp pap refuse`
		Refuses Password Authentication Protocol (PAP) authentication from peers requesting it.
		• If outbound Challenge Handshake Authentication Protocol (CHAP) has been configured (using the **ppp authentication** command), CHAP will be suggested as the authentication method in the refusal packet.
		• PAP authentication is disabled by default.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show running-config`**
	Example:
	`RP/0/RSP0/CPU0:router# show running-config`
		Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

Disabling CHAP Authentication on an Interface

This task explains how to disable CHAP authentication on a serial or POS interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp chap refuse**
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`ppp chap refuse`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ppp chap refuse`
		Refuses CHAP authentication from peers requesting it. After you enter the **ppp chap refuse** command under the specified interface, all attempts by the peer to force the user to authenticate with the help of CHAP are refused.
		• CHAP authentication is disabled by default.
		• If outbound Password Authentication Protocol (PAP) has been configured (using the **ppp authentication** command), PAP will be suggested as the authentication method in the refusal packet.

	Command or Action	Purpose

Step 4	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show running-config`**
	
	Example:
	`RP/0/RSP0/CPU0:router# show running-config`
		Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

Disabling MS-CHAP Authentication on an Interface

This task explains how to disable MS-CHAP authentication on a serial or POS interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp ms-chap refuse**
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/4/0/1`
		Enters interface configuration mode.

Step 3	**`ppp ms-chap refuse`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ppp ms-chap refuse`
		Refuses MS-CHAP authentication from peers requesting it.
		After you enter the **ppp ms-chap refuse** command under the specified interface, all attempts by the peer to force the user to authenticate with the help of MS-CHAP are refused.
		• MS-CHAP authentication is disabled by default.
		• If outbound Password Authentication Protocol (PAP) has been configured (using the **ppp authentication** command), PAP will be suggested as the authentication method in the refusal packet.

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show running-config`**
	Example:
	`RP/0/RSP0/CPU0:router# show running-config`
		Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

Configuring Multilink PPP

This section contains the following procedures:

- Prerequisites, page 604
- Restrictions, page 604
- Configuring the Controller, page 604
- Configuring the Interfaces, page 607
- Configuring MLPPP Optional Features, page 610

Prerequisites

MLPPP and LFI are supported on the 1-Port Channelized OC-3/STM-1 SPA and 2-Port Channelized OC-12/DS0 SPA.

Restrictions

MLPPP for Cisco IOS XR software has the following restrictions:

- Only full rate T1s are supported.
- All links in a bundle must belong to the same SPA.
- All links in a bundle must operate at the same speed.
- A maximum of 10 links per bundle is supported.
- A maximum of 700 bundles per line card is supported.
- A maximum of 2600 bundles per system is supported.
- MLPPP interfaces are not supported with DS0 link members.
- MLPPP interfaces are not be supported with T3 channels as members. Therefore, LFI is also unsupported on T3 channels.
- All serial links in an MLPPP bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MLPPP bundle. The Cisco IOS XR software blocks the following:
  - Attempts to configure a serial interface as a member of an MLPPP bundle if the interface is configured with a nondefault MTU value.
  - Attempts to change the **mtu** command value for a serial interface that is configured as a member of an MLPPP bundle.

In Cisco IOS XR software, multilink processing is controlled by a hardware module called the Multilink Controller, which consists of an ASIC, network processor, and CPU working in conjunction. The MgmtMultilink Controller makes the multilink interfaces behave like the serial interfaces of channelized SPAs.

Configuring the Controller

Perform this task to configure the controller.

SUMMARY STEPS

1. **configure**
2. **controller** *type interface-path-id*
3. **mode** *type*
4. **clock source** {**internal** | **line**}
5. **exit**
6. **controller t1** *interface-path-id*
7. **channel-group** *channel-group-number*
8. **timeslots** *range*
9. **exit**
10. **exit**
11. **controller mgmtmultilink** *interface-path-id*
12. **bundle** *bundle-id*
13. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`controller`** *`type interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0`
		Enters controller configuration submode and specifies the controller name and instance identifier in *rack/slot/module/port* notation.

Step 3	**`mode`** *`type`*
	Example:
	`RP/0/RSP0/CPU0:router# mode t1`
		Configures the type of multilinks to channelize; for example, 28 T1s.

Step 4	**`clock source`** {**`internal`** | **`line`**}
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# clock source internal`
		(Optional) Configures the clocking for the port.
		Note The default clock source is **internal**.

Step 5	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# exit`
		Exits controller configuration mode.

	Command or Action	Purpose

Step 6	**`controller t1`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller t1 0/1/0/0/1`
		Enters T1 configuration mode.

Step 7	**`channel-group`** *`channel-group-number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# channel-group 0`
		Creates a T1 channel group and enters channel group configuration mode for that channel group. Channel group numbers can range from 0 to 23.

Step 8	**`timeslots`** *`range`*
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24`
		Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group.
		• Range is from 1 to 24 time slots.
		Note The time slot range must be from 1 to 24 for the resulting serial interface to be accepted into a MLPPP bundle.

Step 9	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit`
		Exits channel group configuration mode.

Step 10	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-t1)# exit`
		Exits T1 configuration mode and enters global configuration mode.

Step 11	**`controller mgmtmultilink`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller mgmtmultilink 0/1/0/0`
		Enters controller configuration submode for the management of multilink interfaces. Specify the controller name and instance identifier in *rack/slot/module/port* notation.

	Command or Action	Purpose
Step 12	**bundle** *bundle-id* **Example:** `RP/0/RSP0/CPU0:router(config-mgmtmultilink)# bundle 20`	Creates a multilink interface with the specified bundle ID.

Step 13	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Interfaces

Perform this task to configure the interfaces.

Restrictions

- All serial links in an MLPPP bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MLPPP bundle. The Cisco IOS XR software blocks the following:
  - Attempts to configure a serial interface as a member of an MLPPP bundle if the interface is configured with a nondefault MTU value.
  - Attempts to change the **mtu** command value for a serial interface that is configured as a member of an MLPPP bundle.

SUMMARY STEPS

1. **configure**
2. **interface multilink** *interface-path-id*
3. **ipv4 address** *address/mask*
4. **multilink fragment-size** *bytes*
   or
   multilink fragment delay *delay-ms*

5. **keepalive** {*interval* | **disable**}[*retry*]
6. **exit**
7. **interface** *type interface-path-id*
8. **encapsulation** *type*
9. **multilink group** *group-id*
10. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface multilink`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface multilink 0/1/0/0/20`
		Specifies the multilink interface name and instance identifier in *rack/slot/module/port/bundle-id* notation, and enters interface configuration mode.

Step 3	**`ipv4 address`** *`ip-address`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 80.170.0.1/24`
		Assigns an IP address and subnet mask to the interface in the format:
		`A.B.C.D/prefix` `or` *`A.B.C.D/mask`*

Step 4	**multilink fragment-size** *bytes*
	or
	`multilink fragment delay` *`delay-ms`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# multilink fragment-size 350`
	or
	`RP/0/RSP0/CPU0:router(config-if)# multilink fragment delay 2`
		(Optional) Specifies the size of the multilink fragments, such as 128 bytes. Some fragment sizes may not be supported. The default is no fragments.
		or
		(Optional) Specifies the multilink fragment delay in milliseconds. This sets the MLPPP fragment size so that it is equivalent in length to the transmission time delay for any individual member-link (T1s with bandwidths of 1536000bps/192000Bps).
		If the user specifies **fragment delay 2**, the fragment size is (192000*.002)=384B. The usage of this command is exclusive to the usage of **fragment size**. Either command overrides the other.

	Command or Action	Purpose

Step 5	`keepalive {interval | disable}[retry]`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# keepalive disable`
		Sets the keepalive timer for the channel, where:
		• *interval*—Number of seconds (from 1 to 30) between keepalive messages. The default is 10.
		• **disable**—Turns off the keepalive timer.
		• *retry*—(Optional) Number of keepalive messages (from 1 to 255) that can be sent to a peer without a response before transitioning the link to the down state. The default is 3.
		Note To connect with some Cisco IOS devices, multilink keepalives need to be disabled on both devices.

Step 6	`exit`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits interface configuration mode and enters global configuration mode.

Step 7	`interface type interface-path-id`
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface serial 0/1/0/0/1:0`
		Specifies the interface name and instance identifier in *rack/slot/module/port/t1-number:channel-group* notation, and enters interface configuration mode.

Step 8	`encapsulation type`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`
		Specifies the type of encapsulation; in this case, PPP.

Step 9	`multilink group group-id`
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# multilink group 20`
		Specifies the multilink group ID for this interface.

	Command or Action	Purpose

Step 10	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`
		`[cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring MLPPP Optional Features

Perform this task to configure either of the following optional features:

- Minimum number of active links
- Multilink interleave

Note Minimum number active links must be configured at both endpoints.

SUMMARY STEPS

1. **configure**
2. **interface multilink** *interface-path-id*
3. **multilink**
4. **ppp multilink minimum-active links** *value*
5. **multilink interleave**
6. **no shutdown**
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface multilink`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface multilink 0/1/0/0/1`
		Specifies the multilink interface name and instance identifier in *rack/slot/module/port/bundle-id* notation, and enters interface configuration mode.

Step 3	**`multilink`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# multilink`
		Enters interface multilink configuration mode.

Step 4	**`ppp multilink minimum-active links`** *`value`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if-multilink)# ppp multilink minimum-active links 12`
		(Optional) Specifies the minimum number of active links for the multilink interface.
		Note When support for the Noise Attribute is configured to signal PPP to remove links on MLPPP bundles when LNM thresholds are crossed on a link, the links will not be removed below this miminum-active threshold.

Step 5	**`multilink interleave`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if-multilink)# multilink interleave`
		(Optional) Enables interleave on a multilink interface.

	Command or Action	Purpose

Step 6	`no shutdown`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-mutlilink)# no shutdown`
		Removes the shutdown configuration.
		• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state.

Step 7	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config-t3)# end`
	or
	`RP/0/RSP0/CPU0:router(config-t3)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring ICSSO for PPP and MLPPP

This section provides the following ICSSO configuration procedures:

Prerequisites

The Cisco ASR 9000 Series Router supports ICSSO in the following MR-APS, minimum equipment, hardware configurations:

- Two 6-slot or 8-slot chassis
- Four route/switch processors (RSPs), two per chassis (offers a higher degree of reliability)
- Two 20G SIPs, 1 per chassis
- Two of the following SPA types 1 per chassis:
  - 2-Port Channelized OC-12/DS0 SPA

  - 4-Port Channelized T3 SPA
  - 8-Port Channelized T1/E1 SPA
- Two 40 Gigabit Ethernet line cards, 2 per chassis
- Two 4-Port 10 Gigabit Ethernet line cards, 1 per chassis
- 1-Port Channelized OC-3/STM-1 SPA (SPA-1XCHSTM1/OC3)

Restrictions

The following restrictions apply to ICSSO for PPP and MLPPP:

- ICSSO is supported only on two independent routers.
  ICSSO for two line cards on the same router is not supported.
- Automated synchronization or verification of the IOS XR system configuration between the ICSSO peer routers is not available.
- The following restrictions apply to ICSSO on the 2-Port Channelized OC-12/DS0 SPA:
  - ICSSO is supported only on T1/T3 PPP and T1/MLPPP interfaces.
  - T1 member links must terminate on the same SPA.
  - Member links in an MLPPP bundle being protected by MR-APS must all be contained in the same SONET port, this SONET port being a part of the MR-APS protection pair.
  - T1/PPP, T3/PPP and MLPPP encapsulated interfaces on the OC-12 SONET interface can be protected.
- The following restrictions apply to ICSSO on the 1-Port Channelized T3 SPA:
  - Supported for PPP on T3, T1, E1 channels only.
  - Supported for member links in an MLPPP on E1 channels only.
- The following restrictions apply to ICSSO on the 8-Port Channelized T1/E1 SPA:
  - Supported for PPP on T1 and E1 channels only.
  - Supported for member links in an MLPPP on E1 channels only.

Configuring a Basic ICSSO Implementation

Use the following procedure to configure a simple version of ICSSO.

SUMMARY STEPS

1. **config**
2. **redundancy**
3. **multi-router aps**
4. **group** *group_number*
5. **controller sonet** *path*
6. **member ipv4** *address* **backup-interface**
7. **commit**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`config`**
	Example:
	`RP/0/RSP0/CPU0:router# config`
		Enters global configuration mode.

Step 2	**`redundancy`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# redundancy`
		Enters redundancy configuration mode.

Step 3	**`multi-router aps`**
	Example:
	`RP/0/RSP0/CPU0:router(config-redundancy)# multi-router aps`
		Configures Multi-Router APS redundancy and enters APS redundancy configuration mode.

Step 4	**`group`** *`group_number`*
	Example:
	`RP/0/RSP0/CPU0:router(config-redundancy-aps)# group 1`
		Configures the APS redundancy group and assigns the group number.

Step 5	**`controller sonet`** *`path`*
	Example:
	`RP/0/RSP0/CPU0:router(config-redundancy-aps-group)# controller sonet 0/1/0/0`
		Specifies a SONET controller as the APS redundancy backup.

Step 6	**`member ipv4`** *`address`* **`backup-interface type`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config-redundancy-group-controller)# member ipv4 10.10.10.10 backup-interface GigabitEthernet 0/6/0/1`
		Specifies the IP address of the backup interface used by IP-FRR.

Step 7	**`commit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-redundancy-group-controller)# commit`
		Saves the configuration.

Step 8	**`show running config`**
	Example:
	`RP/0/RSP0/CPU0:router#` **`show running config`**
		Displays the current configuration on the router, including MR-APS, SONET controller, and IP address information for verifying the configuration.

Configuring MR-APS

Use the following procedure to configure MR-APS.

SUMMARY STEPS

1. **config**
2. **aps group** *number*
3. **channel {0 | 1} remote** *ip-address*
4. **channel {0 | 1} local sonet** *interface-path-id*
5. **exit**
6. **aps rprplus**
7. **interface GigabitEthernet** *interface-path-id*
8. **description** *text*
9. **ipv4 address** *ipv4-address mask*
10. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**config** **Example:** `RP/0/RSP0/CPU0:router# config`	Enters global configuration mode.
Step 2	**aps group** *number* **Example:** `RP/0/RSP0/CPU0:router(config)# aps group 1`	Adds an automatic protection switching (APS) group and enter APS group configuration mode.
Step 3	**channel {0 | 1} remote** *ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-aps)# channel 0 remote 99.10.1.2`	Assigns a port and interface that is physically located in a remote router as a SONET APS channel. • 0 designates the channel as protect channel. • 1 designates the channel as a working channel.
Step 4	**channel {0 | 1} local sonet** *interface-path-id* Example: `RP/0/RSP0/CPU0:router(config-aps)# channel 1 local SONET 0/1/0/0`	Assigns a local SONET physical port as a SONET APS channel. • 0 designates the channel as protect channel. • 1 designates the channel as a working channel.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-aps)# exit`	Exits to the previous mode.
Step 6	**aps rprplus** Example: `RP/0/RSP0/CPU0:router(config-aps)# aps rprplus`	Extends the APS hold timer for a switchover.

	Command or Action	Purpose
Step 7	**interface GigabitEthernet** *interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/6/0/0`	Creates a Gigabit Ethernet interface as the path to the MR-APS peer, and enters interface configuration mode.
Step 8	**description** *text* **Example:** `RP/0/RSP0/CPU0:router(config-if)# description MR-APS PGP interface for aps group 1`	Adds a text description to this interface.
Step 9	**ipv4 address** *ipv4-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if )# ipv4 address 99.10.1.1 255.255.255.0`	Sets the primary IPv4 address and subnet mask for an interface.
Step 10	**commit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# commit`	Saves the current configuration.

Configuring SSRP on Serial and Multilink Interfaces

Use the following procedure to configure SSRP on serial and multilink interfaces:

SUMMARY STEPS

1. **config**
2. **ssrp profile** *profile-name*
3. **peer ipv4 address** *A.B.C.D*
4. **exit**
5. **ssrp location** *node_id*
6. **group** *group-id* **profile** *profile_name*
7. **group** *group-id* **profile** *profile_name*
8. **exit**
9. **interface serial** *interface-path-id*
10. **ssrp group** *group-number* **id** *id-number* **ppp**
11. **encapsulation ppp**
12. **multilink**
13. **group** *group-id*
14. **exit**
15. **keepalive disable**
16. **exit**

17. **interface serial** *interface-path-id*
18. **ssrp group** *group-number* **id** *id-number* **ppp**
19. **encapsulation ppp**
20. **multilink**
21. **group** *group-id*
22. **exit**
23. **keepalive disable**
24. **exit**
25. **interface multilink interface-path-id**
26. **ipv4 address** *ipv4-address mask*
27. **ssrp group** *group-number* **id** *id-number* **ppp**
28. **encapsulation ppp**
29. **shutdown**
30. **keepalive disable**
31. **exit**
32. **controller MgmtMultilink** *interface-path-id*
33. **bundle** *bundleID*
34. **bundle** *bundleID*
35. **commit**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`config`**
	Example:
	`RP/0/RSP0/CPU0:router# config`
		Enters global configuration mode.

Step 2	**`ssrp profile`** *`profile-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# ssrp profile Profile_1`
		Configures the Session State Redundancy Protocol (SSRP) profile and enters the SSRP configuration mode.

Step 3	**`peer ipv4 address`** *`A.B.C.D`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# peer ipv4 address 10.10.10.10`
		Configures the IPv4 address for a Session State Redundancy Protocol (SSRP) peer.

Step 4	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-aps)# exit`
		Exits to the previous mode.

	Command or Action	Purpose
Step 5	**ssrp location** *node_id* **Example:** `RP/0/RSP0/CPU0:router(config)# ssrp location 0/1/CPU0`	Specifies the node on which to create a Session State Redundancy Protocol (SSRP) group and enters the SSRP node configuration mode
Step 6	**group** *group-id* **profile** *profile_name* **Example:** `RP/0/RSP0/CPU0:router(config-ssrp)# group 1 profile Profile_1`	Creates a Session State Redundancy Protocol (SSRP) group and associates it with a profile.
Step 7	**group** *group-id* **profile** *profile_name* **Example:** `RP/0/RSP0/CPU0:router(config-ssrp-node)# group 2 profile Profile_2`	Creates a second Session State Redundancy Protocol (SSRP) group and associates it with a profile.
Step 8	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ssrp-node)# exit`	Exits to the previous mode.
Step 9	**interface serial** *interface-path-id[.subinterface]* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/1/0/0/1/1:0`	Physical interface or virtual interface. **Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router. For more information about the syntax for the router, use the question mark (**?**) online help function.
Step 10	**ssrp group** *group-number* **id** *id-number* **ppp** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ssrp group 1 id 1 ppp`	Attaches an SSRP group on the interface.
Step 11	**encapsulation ppp** **Example:** `RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`	Enables encapsulation for communication with routers using the Point-to-Point Protocol (PPP).
Step 12	**multilink** **Example:** `RP/0/RSP0/CPU0:router(config-if)# multilink`	Enters the multilink interface configuration mode.
Step 13	**group** *group-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# group 1`	Attaches a Session State Redundancy Protocol (SSRP) group to this interface.
Step 14	**exit** **Example:** `RP/0/RSP0/CPU0:router(config)# exit`	Exits to the previous mode.

	Command or Action	Purpose
Step 15	**keepalive disable** **Example:** `RP/0/RSP0/CPU0:router(config)# keepalive disable`	Disables the keepalive timer for this interface.
Step 16	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# exit`	Exits to the previous mode.
Step 17	**interface serial** *interface-path-id[.subinterface]* **Example:** `RP/0/RSP0/CPU0:router(config)# interface serial 0/1/0/0/1/2:0`	Physical interface or virtual interface. **Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router. For more information about the syntax for the router, use the question mark (**?**) online help function.
Step 18	**ssrp group** *group-number* **id** *id-number* **ppp** **Example:** `RP/0/RSP0/CPU0:router(config-if)# ssrp group 1 id 2 ppp`	Attaches an SSRP group on the interface.
Step 19	**encapsulation ppp** **Example:** `RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`	Enables encapsulation for communication with routers using the Point-to-Point Protocol (PPP).
Step 20	**multilink** **Example:** `RP/0/RSP0/CPU0:router(config-if)# multilink`	Enters the multilink interface configuration mode.
Step 21	**group** *group-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# group 1`	Attaches a Session State Redundancy Protocol (SSRP) group to this interface.
Step 22	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# exit`	Exits to the previous mode.
Step 23	**keepalive disable** **Example:** `RP/0/RSP0/CPU0:router(config-if)# keepalive disable`	Disables the keepalive timer for this interface.
Step 24	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# exit`	Exits to the previous mode.

	Command or Action	Purpose

Step 25	**`interface multilink`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface Multilink 0/1/0/0/1`
		Physical interface or virtual interface.
		Note Use the **show interfaces** command to see a list of all interfaces currently configured on the router.
		For more information about the syntax for the router, use the question mark (**?**) online help function.

Step 26	**`ipv4 address`** *`ipv4-address mask`*
	Example:
	`RP/0/RSP0/CPU0:router(config-if )# ipv4 address 10.10.10.10 255.255.255.0`
		Sets the primary IPv4 address and subnet mask for an interface.

Step 27	**`ssrp group`** *`group-number`* **`id`** *`id-number`* **`ppp`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# ssrp group 1 id 3 ppp`
		Attaches an SSRP group on the interface.

Step 28	**`encapsulation ppp`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp`
		Enables encapsulation for communication with routers using the Point-to-Point Protocol (PPP).

Step 29	**`shutdown`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# shutdown`
		Brings the interface administratively down for configuration.

Step 30	**`keepalive disable`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# keepalive disable`
		Disables the keepalive timer for this interface.

Step 31	**`exit`**
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# exit`
		Exits to the previous mode.

Step 32	**`controller MgmtMultilink`** *`interface-path-id`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# controller MgmtMultilink 0/1/0/0`
		Configure a controller for a generic multilink bundle and enters MgmtMultilink configuration mode.

Step 33	**`bundle`** *`bundleID`*
	Example:
	`RP/0/RSP0/CPU0:router(config-mgmtmultilink)# bundle 1`
		Creates a multilink interface bundle.

	Command or Action	Purpose
Step 34	**bundle** *bundleID* **Example:** `RP/0/RSP0/CPU0:router(config-mgmtmultilink)# bundle 2`	Creates a multilink interface bundle.
Step 35	**commit** **Example:** `RP/0/RSP0/CPU0:router(config-mgmtmultilink)# commit`	Saves the current configuration.

Configuration Examples for PPP

This section provides the following configuration examples:

- Configuring a POS Interface with PPP Encapsulation: Example, page 621
- Configuring a Serial Interface with PPP Encapsulation: Example, page 621
- ICSSO for PPP and MLPPP Configuration: Examples, page 622
- Verifying Multilink PPP Configurations, page 629

Configuring a POS Interface with PPP Encapsulation: Example

The following example shows how to create and configure a POS interface with PPP encapsulation:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# ppp pap sent-username P1_TEST-8 password xxxx
RP/0/RSP0/CPU0:router(config-if)# ppp authentication chap pap MIS-access
RP/0/RSP0/CPU0:router(config-if)# ppp chap password encrypted xxxx
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure POS interface 0/3/0/1 to allow two additional retries after an initial authentication failure (for a total of three failed authentication attempts):

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface POS 0/3/0/1
RP/0/RSP0/CPU0:router(config-if)# ppp max-bad-auth 3
```

Configuring a Serial Interface with PPP Encapsulation: Example

The following example shows how to create and configure a serial interface with PPP MS-CHAP encapsulation:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
```

```
RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RSP0/CPU0:router(config-if)# no shutdown
RP/0/RSP0/CPU0:router(config-if)# ppp authentication ms-chap MIS-access
RP/0/RSP0/CPU0:router(config-if)# ppp ms-chap password encrypted xxxx
RP/0/RSP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

Configuring MLPPP: Example

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# controller t3 0/1/0/0/1
RP/0/RSP0/CPU0:router# mode t1
RP/0/RSP0/CPU0:router(config-t3)# clock source internal
RP/0/RSP0/CPU0:router(config-t3)# exit
RP/0/RSP0/CPU0:router(config)# controller t1 0/1/0/0/1/1
RP/0/RSP0/CPU0:router(config-t1)# channel-group 0
RP/0/RSP0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/RSP0/CPU0:router(config-t1-channel_group)# exit
RP/0/RSP0/CPU0:router(config-t1)# exit
RP/0/RSP0/CPU0:router(config)# controller mgmtmultilink 0/1/0/0
RP/0/RSP0/CPU0:router(config-mgmtmultilink)# bundle 20
RP/0/RSP0/CPU0:router(config-t3)# commit
RP/0/RSP0/CPU0:router(config-t3)# exit

RP/0/RSP0/CPU0:router(config)# interface multilink 0/1/0/0/20
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 80.170.0.1/24
RP/0/RSP0/CPU0:router(config-if)# multilink fragment-size 128
RP/0/RSP0/CPU0:router(config-if)# keepalive disable
RP/0/RSP0/CPU0:router(config-if)# exit
RP/0/RSP0/CPU0:router(config)# interface serial 0/1/0/0/1/1:0
RP/0/RSP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RSP0/CPU0:router(config-if)# multilink group 20
RP/0/RSP0/CPU0:router(config-t3)# commit
RP/0/RSP0/CPU0:router(config-t3)# exit

RP/0/RSP0/CPU0:router(config)# interface multilink 0/1/0/0/1
RP/0/RSP0/CPU0:router(config-if)# multilink
RP/0/RSP0/CPU0:router(config-if-multilink)# ppp multilink minimum-active links 10
RP/0/RSP0/CPU0:router(config-if-multilink)# multilink interleave
RP/0/RSP0/CPU0:router(config-if-mutlilink)# no shutdown
RP/0/RSP0/CPU0:router(config-t3)# commit
```

ICSSO for PPP and MLPPP Configuration: Examples

This section provides the following examples of ICSSO configuration and related configurations:

- ICSSO Configuration: Example, page 623
- Channelized SONET Controller Configuration for Use with ICSSO: Example, page 623
- MR-APS Configuration: Example, page 623
- SSRP on Serial and Multilink Interfaces Configuration: Example, page 624
- VRF on Multilink Configuration for Use with ICSSO: Example, page 625
- VRF on Ethernet Configuration for Use with ICSSO: Example, page 625
- OSPF Configuration for Use with ICSSO: Example, page 626
- Verifying ICSSO Configuration: Examples, page 626

ICSSO Configuration: Example

The following example shows how to configure ICSSO on a SONET controller:

```
config
    redundancy
        multi-router aps
        group 1
        controller sonet 0/1/0/0
            member ipv4 10.10.10.10 backup-interface GigabitEthernet 0/6/0/1
            commit
show running config
```

Channelized SONET Controller Configuration for Use with ICSSO: Example

The following example shows how to configure channelized SONET controllers for use with ICSSO:

```
config
    controller SONET0/7/1/0
        framing sonet
        sts 1
        mode t3
!
        sts 2
        mode t3
!
        sts 3
        mode t3
!
    controller T3 0/7/0/1
        mode t1
        framing auto-detect
!
    controller T1 0/7/0/1/1
        channel-group 0
        timeslots 1-24
```

MR-APS Configuration: Example

The following example shows how to configure MR-APS:

```
config
    aps group 1
        channel 0 remote 99.10.1.2
        channel 1 local SONET0/1/0/0
!
    aps rprplus
!
    interface GigabitEthernet0/6/0/0
        description MR-APS PGP interface for aps group 1
        ipv4 address 99.10.1.1 255.255.255.0
```

The following example shows how to configure a redundancy group manager:

```
// mr-aps part:
aps group 1
 channel 0 remote 99.10.1.2
 channel 1 local SONET0/1/0/0
!
// ssrp part:
```

```
ssrp location 0/1/CPU0
 group 1 profile TEST
!
ssrp profile TEST
 peer ipv4 address 99.10.1.2
!
// redundancy group manager part:
redundancy
 multi-router aps
  group 1
   controller SONET0/1/0/0
    member ipv4 99.30.1.2 backup-interface GigabitEthernet0/6/0/4
   !

// ospf part:
router ospf 1
 nsr
 nsf ietf
 redistribute connected instance IPCP
 redistribute static
 area 0
  interface GigabitEthernet0/6/0/4
  !
 !
!

show redundancy-group multi-router aps
```

SSRP on Serial and Multilink Interfaces Configuration: Example

The following example shows how to configure SSRP on serial interfaces with PPP encapsulation and multilink interfaces:

```
config
    ssrp profile TEST
        peer ipv4 address 99.10.1.2
!
    ssrp location 0/1/CPU0
        group 1 profile TEST
!
    interface Serial0/1/0/0/1/1:0
        ssrp group 1 id 1 ppp
        encapsulation ppp
        multilink
        group 1
!
    keepalive disable
!
    interface Serial0/1/0/0/1/2:0
        ssrp group 1 id 2 ppp
        encapsulation ppp
        multilink
        group 1
!
    keepalive disable
!
    interface Multilink0/1/0/0/1
        ipv4 address 51.1.1.1 255.255.255.0
        ssrp group 1 id 3 ppp
        encapsulation ppp
        shutdown
!
keepalive disable
!
    controller MgmtMultilink0/1/0/0
        bundle 1
```

Note For more information on configuring serial interfaces, refer to the *Configuring Serial Interfaces on the Cisco ASR 9000 Series Router* module of this document.

Note For more information on configuring Multilink, refer to Configuring Multilink PPP, page 604.

VRF on Multilink Configuration for Use with ICSSO: Example

The following example shows how to configure VPN Routing and Forwarding (VRF) on a Multilink interface for use with ICSSO:

```
config
    vrf EvDO-vrf
        address-family ipv4 unicast
!
    interface Multilink 0/0/0/0/1
        description To EvDO BTS Number 1
        vrf EvDO-vrf
        ipv4 address 150.0.1.3 255.255.255.0
        encapsulation ppp
!
```

Note For more information on configuring VRF, refer to the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*. For more information on configuring Multilink, refer to Configuring Multilink PPP, page 604.

VRF on Ethernet Configuration for Use with ICSSO: Example

The following example shows how to configure VPN Routing and Forwarding (VRF) on an Ethernet interface for use with ICSSO:

```
config
    vrf EvDO-vrf
        address-family ipv4 unicast
!
    interface GigabitEthernet 1/0/0/0.20
        description Inter-ASR9000 EvDO VLAN
        vrf EvDO-vrf
        encapsulation dot1q 20
```

Note For more information on configuring VRF, refer to the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*. For more information on configuring Ethernet, refer to the Configuring Ethernet OAM on the Cisco ASR 9000 Series Router module of this document.

OSPF Configuration for Use with ICSSO: Example

Aggregation routers that terminate PPP sessions to a set of cell sites, advertise their availability to LAN switches using Open Shortest Path First (OSPF). The following example shows how to configure OSPF for use with ICSSO:

```
config
   router ospf 1
      nsr
      nsf ietf
      redistribute connected instance IPCP
      redistribute static
      area 0
   interface GigabitEthernet 0/6/0/1
!
```

Note For more information on configuring OSPF, refer to the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*.

Verifying ICSSO Configuration: Examples

The following examples show how to verify ICSSO configuration:

- Verifying SSRP Groups: Example, page 626
- Verifying ICSSO Status: Example, page 627
- Verifying MR-APS Configuration: Example, page 627
- Verifying OSPF Configuration: Example, page 628

Verifying SSRP Groups: Example

The following example shows how to verify SSRP Group configuration:

```
RP/0/RSP0/CPU0:Router# show ssrp groups all det loc 0/1/cpu0

Tue Nov 10 16:57:55.911 UTC

Group ID: 1
  Conn (ACT,SB): UP,UP
  Profile:  TEST
  Peer:     99.10.1.2
  Max-hops: 255
  Sessions: 3
  Channels Created
  Client:            PPP
    Active Init:      TRUE
    Standby Init:     TRUE
    Active State:     IDT-End-Sent
    Standby State:    IDT-End-Received
    Auth-Req Pending: FALSE
    Active   ID Out:             93
    Active   ID In:              93
    Active   Last Reply In:      93
    Active   Counter:             5

    Standby  ID Out:             50
    Standby  ID In:              50
    Standby  Last Reply In:           50
    Standby  Counter:                  5

    Session   Interface
    -----------------------------
    1         Se0/1/0/0/1/1:0
    2         Se0/1/0/0/1/2:0
    3         Mu0/1/0/0/1
```

Verifying ICSSO Status: Example

The following example shows how to verify ICSSO status:

```
RP/0/RSP0/CPU0:Router# show ppp sso sum loc 0/1/cpu0
Tue Nov 10 16:59:00.253 UTC

Not-Ready    : The session is not yet ready to run as Active or Standby
Stby-UnNegd  : In Standby mode, no replication state received yet
Act-Down     : In Active mode, lower layer not yet up
Deactivating : Session was Active, now going Standby
Act-UnNegd   : In Active mode, not fully negotiated yet
Stby-Negd    : In Standby mode, replication state received and pre-programmed
Activating   : Session was Standby and pre-programmed, now going Active
Act-Negd     : In Active mode, fully negotiated and up
-            : This layer not running

                      Not-   Stby-  Act- Deactiv- Act-   Stby-  Activ- Act-
Layer        | Total  Ready  UnNegd Down   ating  UnNegd Negd   ating  Negd
-------------+------- ------ ------ ------ ------ ------ ------ ------ ------
LCP          |      6      0      0      0      0      0      0      0      6
of-us-auth   |      6      0      0      0      0      0      0      0      6
of-peer-auth |      6      0      0      0      0      0      0      0      6
IPCP         |      2      0      0      0      0      0      0      0      2
```

Verifying MR-APS Configuration: Example

The following examples show how to verify MR-APS configuration:

Example 1:

```
RP/0/RSP0/CPU0:Router# show redundancy-group multi-router aps all

Tue Nov 10 17:00:14.018 UTC

Interchassis Group: 1
             State: FRR ADD SENT
        Controller: SONET0/1/0/0                        0x2000080
  Backup Interface: GigabitEthernet0/6/0/1              0x10000180
  Next Hop IP Addr: 10.10.10.10

Interchassis Group: Not Configured
             State: WAIT CONFIG
        Controller: SONET0/1/0/1                        0x20003c0
  Backup Interface: None                                0x0
  Next Hop IP Addr: 0.0.0.0
```

Example 2:

```
RP/0/RSP0/CPU0:Router# show cef adj rem loc 0/6/cpu0

Tue Nov 10 17:00:30.471 UTC
Display protocol is ipv4
Interface    Address                                         Type    Refcount

SO0/1/0/0    Ifhandle: 0x2000080                             remote  2
             Adjacency: PT:0xa47c9cf4
             Interface: SO0/1/0/0
             Interface Type: 0x0, Base Flags: 0x110000 (0xa4a00494)
             Nhinfo PT: 0xa4a00494, Idb PT: 0xa4cd60d8, If Handle: 0x2000080
             Ancestor If Handle: 0x0

             Protect FRR: 0xa4a8a040
             Backup FRR: 0xa4a89f34
             Backup NH: 0xa4a00a74
             Backup IFH: 0x10000180
             Backup Interface: Gi0/6/0/1
             Backup IP: 10.10.10.10

             FRR Active: 0
```

Verifying OSPF Configuration: Example

The following examples show how to verify OSPF configuration:

Example 1:

```
RP/0/RSP0/CPU0:Router# show route back
Tue Nov 10 17:01:48.974 UTC

Codes: C - connected, S - static, R - RIP, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR
       A - access/subscriber

C    51.1.1.2/32 is directly connected, 00:10:03, Multilink0/1/0/0/1
                 Backup  O E2 [110/20] via 10.10.10.10, GigabitEthernet0/6/0/1
C    52.1.1.2/32 is directly connected, 00:11:47, Multilink0/1/0/0/2
                 Backup  O E2 [110/20] via 10.10.10.10, GigabitEthernet0/6/0/1
S    110.0.0.2/32 [1/0] via 51.1.1.2, 00:11:40
                 Backup  O E2 [110/20] via 10.10.10.10, GigabitEthernet0/6/0/1
```

Example 2:

```
RP/0/RSP0/CPU0:Router# show route 51.1.1.2
Tue Nov 10 17:02:26.507 UTC

Routing entry for 51.1.1.2/32
  Known via "connected IPCP", distance 0, metric 0 (connected)
  Installed Nov 10 16:51:45.703 for 00:10:40
  Routing Descriptor Blocks
    51.1.1.2 directly connected, via Multilink0/1/0/0/1
      Route metric is 0
  No advertising protos.
```

Verifying Multilink PPP Configurations

Use the following show commands to verify and troubleshoot your multilink configurations:

- show multilink interfaces: Examples, page 629
- show ppp interfaces multilink: Example, page 631
- show ppp interface serial: Example, page 632
- show imds interface multilink: Example, page 632

show multilink interfaces: Examples

```
RP/0/RSP0/CPU0:Router# show multilink interfaces Serial 0/4/3/1/10:0
Mon Sep 21 09:24:19.604 UTC

Serial0/4/3/1/10:0 is up, line protocol is up
      Encapsulation: PPP
      Multilink group id: 6
      Member status: ACTIVE

RP/0/RSP0/CPU0:Router# show multilink interfaces Multilink 0/4/3/0/3
Mon Sep 21 09:17:12.131 UTC

Multilink0/4/3/0/3 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 1 active, 1 inactive
    - Serial0/4/3/1/5:0 is up, line protocol is up
      Encapsulation: PPP
      Multilink group id: 3
      Member status: ACTIVE

  - Serial0/4/3/1/6:0 is administratively down, line protocol is administratively down
Encapsulation: PPP
      Multilink group id: 3
      Member status: INACTIVE : LCP has not been negotiated

  Fragmentation Statistics
  Input Fragmented packets 0             Input Fragmented bytes 0
  Output Fragmented packets 0            Output Fragmented bytes 0
  Input Unfragmented packets 0           Input Unfragmented bytes 0
  Output Unfragmented packets 0          Output Unfragmented bytes 0
  Input Reassembled packets 0            Input Reassembled bytes 0

RP/0/5/CPU0:Mav-IOX-Rahul#sho multilink interfaces Serial 0/4/3/1/10:0
Mon Sep 21 09:24:19.604 UTC

Serial0/4/3/1/10:0 is up, line protocol is up
      Encapsulation: PPP
      Multilink group id: 6
      Member status: ACTIVE

RP/0/RSP0/CPU0:Router# show multilink interfaces
Mon Sep 21 09:15:10.679 UTC

Multilink0/4/3/0/1 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: FR
  Member Links: 1 active, 1 inactive
   - Serial0/4/3/1/2:0: INACTIVE : Down  (Member link idle)
   - Serial0/4/3/1/1:0: ACTIVE : Up


Multilink0/4/3/0/10 is up, line protocol is down
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 0 active, 0 inactive
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/100 is administratively down, line protocol is administratively down
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 0 active, 0 inactive
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/2 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: FR
  Member Links: 2 active, 0 inactive
   - Serial0/4/3/1/4:0: ACTIVE : Up
   - Serial0/4/3/1/3:0: ACTIVE : Up


Multilink0/4/3/0/3 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 1 active, 1 inactive
   - Serial0/4/3/1/5:0: ACTIVE
   - Serial0/4/3/1/6:0: INACTIVE : LCP has not been negotiated
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/4 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 2 active, 0 inactive
   - Serial0/4/3/1/8:0: ACTIVE
   - Serial0/4/3/1/7:0: ACTIVE
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/5 is up, line protocol is up
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 1 active, 0 inactive
   - Serial0/4/3/1/9:0: ACTIVE
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/6 is up, line protocol is up
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 1 active, 0 inactive
   - Serial0/4/3/1/10:0: ACTIVE
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/7 is up, line protocol is down
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 0 active, 1 inactive
   - Serial0/4/3/1/11:0: INACTIVE : LCP has not been negotiated
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/8 is up, line protocol is down
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 0 active, 1 inactive
   - Serial0/4/3/1/12:0: INACTIVE : LCP has not been negotiated
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0
```

show ppp interfaces multilink: Example

```
RP/0/RSP0/CPU0:Router# show ppp interfaces multilink 0/3/1/0/1

Multilink 0/3/1/0/1 is up, line protocol is up
LCP: Open
     Keepalives disabled
```

```
  IPCP: Open
     Local IPv4 address: 1.1.1.2
     Peer IPv4 address:  1.1.1.1
  Multilink
     Member Links: 2 active, 1 inactive (min-active 1)
      - Serial0/3/1/0/0:0: ACTIVE
      - Serial0/3/1/0/1:0: ACTIVE
      - Serial0/3/1/0/2:0: INACTIVE : LCP has not been negotiated
```

show ppp interface serial: Example

```
RP/0/RSP0/CPU0:Router# show ppp interface Serial 0/3/1/0/0:0

Serial 0/3/1/0/0:0 is up, line protocol is up
  LCP: Open
     Keepalives disabled
     Local MRU: 1500 bytes
     Peer  MRU: 1500 bytes
     Local Bundle MRRU: 1596 bytes
     Peer  Bundle MRRU: 1500 bytes
     Local Endpoint Discriminator: 1b61950e3e9ce8172c8289df0000003900000001
     Peer  Endpoint Discriminator: 7d046cd8390a4519087aefb90000003900000001
  Authentication
     Of Peer: <None>
     Of Us:   <None>
  Multilink
     Multilink group id: 1
     Member status: ACTIVE
```

show imds interface multilink: Example

```
RP/0/RSP0/CPU0:Router# show imds interface Multilink 0/3/1/0/1

IMDS INTERFACE DATA (Node 0x0)

Multilink0_3_1_0_1 (0x04001200)
-----------------------
flags: 0x0001002f    type: 55 (IFT_MULTILINK)    encap: 52 (ppp)
state: 3 (up)     mtu: 1600    protocol count: 3
control parent: 0x04000800     data parent: 0x00000000
       protocol         capsulation           state        mtu
    --------------- -------------------- --------------- --------
    12 (ipv4)
                    26 (ipv4)            3 (up)          1500
                    47 (ipcp)            3 (up)          1500
    16 (ppp_ctrl)
                    53 (ppp_ctrl)        3 (up)          1500
    0 (Unknown)
                    139 (c_shim)         3 (up)          1600
                    52 (ppp)             3 (up)          1504
                    56 (queue_fifo)      3 (up)          1600
                    60 (txm_nopull)      3 (up)          1600
```

Additional References

The following sections provide references related to PPP encapsulation.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Cisco IOS XR interface configuration commands	*Cisco IOS XR Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a router using Cisco IOS XR software	*Cisco IOS XR Getting Started Guide*
Cisco IOS XR AAA services configuration information	*Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC-1661	*The Point-to-Point Protocol (PPP)*
RFC- 1994	*PPP Challenge Handshake Authentication Protocol (CHAP)*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring 802.1Q VLAN Interfaces on the Cisco ASR 9000 Series Router

This module describes the configuration and management of 802.1Q VLAN interfaces on the Cisco ASR 9000 Series Aggregation Services Routers.

The IEEE 802.1Q specification establishes a standard method for tagging Ethernet frames with VLAN membership information, and defines the operation of VLAN bridges that permit the definition, operation, and administration of VLAN topologies within a bridged LAN infrastructure.

The 802.1Q standard is intended to address the problem of how to divide large networks into smaller parts so broadcast and multicast traffic does not use more bandwidth than necessary. The standard also helps provide a higher level of security between segments of internal networks.

Feature History for Configuring 802.1Q VLAN Interfaces

Release	Modification
Release 3.7.2	This feature was introduced on the Cisco ASR 9000 Series Router.
Release 3.9.0	Layer 2 dot1q was updated. Encapsulation dot1q was added.

Contents

Prerequisites for Configuring 802.1Q VLAN Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring 802.1Q VLAN interfaces, be sure that the following conditions are met:

- You must have configured a Gigabit Ethernet interface, a 10-Gigabit Ethernet interface, or an Ethernet bundle interface.

Information About Configuring 802.1Q VLAN Interfaces

To configure 802.1Q VLAN interfaces, you must understand the following concepts:

- 802.1Q VLAN Overview, page 636
- 802.1Q Tagged Frames, page 636
- CFM on 802.1Q VLAN Interfaces, page 637
- Subinterfaces, page 637
- Subinterface MTU, page 637
- Native VLAN, page 637
- EFPs, page 637
- Layer 2 VPN on VLANs, page 638
- Other Layer 2 VPN Features, page 639

802.1Q VLAN Overview

A VLAN is a group of devices on one or more LANs that are configured so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments. Because VLANs are based on logical instead of physical connections, they are very flexible for user and host management, bandwidth allocation, and resource optimization.

The IEEE 802.1Q protocol standard addresses the problem of dividing large networks into smaller parts so broadcast and multicast traffic does not consume more bandwidth than necessary. The standard also helps provide a higher level of security between segments of internal networks.

The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames.

Cisco IOS XR software supports VLAN subinterface configuration on Gigabit Ethernet and10-Gigabit Ethernet interfaces.

802.1Q Tagged Frames

The IEEE 802.1Q tag-based VLAN uses an extra tag in the MAC header to identify the VLAN membership of a frame across bridges. This tag is used for VLAN and quality of service (QoS) priority identification. The VLANs can be created statically by manual entry or dynamically through Generic Attribute Registration Protocol (GARP) VLAN Registration Protocol (GVRP). The VLAN ID associates a frame with a specific VLAN and provides the information that switches must process the frame across the network. A tagged frame is four bytes longer than an untagged frame and contains two bytes of Tag Protocol Identifier (TPID) residing within the type and length field of the Ethernet frame and two bytes of Tag Control Information (TCI) which starts after the source address field of the Ethernet frame.

CFM on 802.1Q VLAN Interfaces

Configuring Connectivity Fault Management (CFM) for monitoring 802.1Q VLAN interfaces is identical to configuring CFM for monitoring Ethernet interfaces.

For information on configuring CFM for Ethernet interfaces, refer to the following sections in the Configuring Ethernet OAM on the Cisco ASR 9000 Series Router module:

- Ethernet CFM, page 70
- Configuring Ethernet CFM, page 108
- Ethernet CFM Service Configuration: Example, page 164
- Ethernet CFM Show Command: Examples, page 165

Subinterfaces

Subinterfaces are logical interfaces created on a hardware interface. These software-defined interfaces allow for segregation of traffic into separate logical channels on a single hardware interface as well as allowing for better utilization of the available bandwidth on the physical interface.

Subinterfaces are distinguished from one another by adding an extension on the end of the interface name and designation. For instance, the Ethernet subinterface 23 on the physical interface designated TenGigE 0/1/0/0 would be indicated by TenGigE 0/1/0/0.23.

Before a subinterface is allowed to pass traffic it must have a valid tagging protocol encapsulation and VLAN identifier assigned. All Ethernet subinterfaces always default to the 802.1Q VLAN encapsulation. However, the VLAN identifier must be explicitly defined.

Subinterface MTU

The subinterface maximum transmission unit (MTU) is inherited from the physical interface with an additional four bytes allowed for the 802.1Q VLAN tag.

Native VLAN

The Cisco ASR 9000 Series Router does not support a native VLAN. However, the equivalent functionality is accomplished using an **encapsulation** command as follows:

```
encapsulation dot1q TAG-ID, untagged
```

EFPs

An Ethernet Flow Point (EFP) is a Metro Ethernet Forum (MEF) term describing abstract router architecture. On the Cisco ASR 9000 Series Router, an EFP is implemented by an L2 subinterface with a VLAN encapsulation. The term EFP is used synonymously with an VLAN tagged L2 subinterface.

Layer 2 VPN on VLANs

The Layer 2 Virtual Private Network (L2VPN) feature enables Service Providers (SPs) to provide Layer 2 services to geographically disparate customer sites.

The configuration model for configuring VLAN attachment circuits (ACs) is similar to the model used for configuring basic VLANs, where the user first creates a VLAN subinterface, and then configures that VLAN in subinterface configuration mode. To create an AC, you need to include the **l2transport** keyword in the **interface** command string to specify that the interface is a Layer 2 interface.

VLAN ACs support three modes of L2VPN operation:

- Basic Dot1Q AC—The AC covers all frames that are received and sent with a specific VLAN tag.
- QinQ AC—The AC covers all frames received and sent with a specific outer VLAN tag and a specific inner VLAN tag. QinQ is an extension to Dot1Q that uses a stack of two tags.
- Q-in-Any AC—The AC covers all frames received and sent with a specific outer VLAN tag and any inner VLAN tag, as long as that inner VLAN tag is not L3 terminated. Q-in-Any is an extension to QinQ that uses wildcarding to match any second tag.

Note The Q-in-Any mode is a variation of the basic Dot1Q mode. In Q-in-Any mode, the frames have a basic QinQ encapsulation; however, in Q-in-Any mode the inner tag is not relevant, except for the fact that a few specific inner VLAN tags are siphoned for specific services. For example, a tag may be used to provide L3 services for general internet access.

Each VLAN on a CE-to-PE link can be configured as a separate L2VPN connection (using either VC type 4 or VC type 5). To configure L2VPN on VLANs, see the "Configuring an Attachment Circuit on a VLAN" section on page 641.

Keep the following in mind when configuring L2VPN on a VLAN:

- Cisco IOS XR software supports 4k ACs per LC.
- In a point-to-point connection, the two ACs do not have to be of the same type. For example, a port mode Ethernet AC can be connected to a Dot1Q Ethernet AC.
- Pseudowires can run in VLAN mode or in port mode. A pseudowire running in VLAN mode has a single Dot1Q tag, while a pseudo-wire running in port mode has no tags. Some interworking is required to connect these different types of circuits together. This interworking takes the form of popping, pushing, and rewriting tags. The advantage of Layer 2 VPN is that is simplifies the interworking required to connect completely different media types together.
- The ACs on either side of an MPLS pseudowire can be different types. In this case, the appropriate conversion is carried out at one or both ends of the AC to pseudowire connection.

Use the **show interfaces** command to display AC and pseudowire information.

Note For detailed information about configuring an L2VPN network, see the "*Implementing MPLS Layer 2* VPNs" module of the *Cisco ASR 9000 Series Router Multiprotocol Label Switching Configuration Guide.*

Other Layer 2 VPN Features

For information on the following Layer 2 VPN features, refer to the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* and the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*:

- Provider Backbone Bridge (PBB) 802.1ah
- Policy-Based Forwarding (PBF)
- MVRP 802.1 (MVRP-lite)

How to Configure 802.1Q VLAN Interfaces

This section contains the following procedures:

- Configuring 802.1Q VLAN Subinterfaces, page 639
- Configuring an Attachment Circuit on a VLAN, page 641
- Removing an 802.1Q VLAN Subinterface, page 643

Configuring 802.1Q VLAN Subinterfaces

This task explains how to configure 802.1Q VLAN subinterfaces. To remove these subinterfaces, see the "Removing an 802.1Q VLAN Subinterface" section of this module.

SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE** |**Bundle-Ether**} *interface-path-id.subinterface*
3. **encapsulation dot1q**
4. **ipv4 address** *ip-address mask*
5. **exit**
6. Repeat Step 2 through Step 5 to define the rest of the VLAN subinterfaces.
7. **end**
   or
   commit
8. **show ethernet trunk bundle-ether** *instance* (Optional)

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** {**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *interface-path-id.subinterface* **Example:** `RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/2/0/4.10`	Enters subinterface configuration mode and specifies the interface type, location, and subinterface number. • Replace the *interface-path-id* argument with one of the following instances: – Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack/slot/module/port*, and a slash between values is required as part of the notation. – Ethernet bundle instance. Range is from 1 through 65535. • Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095. • Naming notation is *interface-path-id.subinterface*, and a period between arguments is required as part of the notation.
Step 3	**encapsulation dot1q** **Example:** `RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100, untagged`	Sets the Layer 2 encapsulation of an interface. **Note** The **dot1q vlan** command is replaced by the **encapsulation dot1q** command on the Cisco ASR 9000 Series Router. It is still available for backward-compatibility, but only for Layer 3 interfaces.
Step 4	**ipv4 address** *ip-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 178.18.169.23/24`	Assigns an IP address and subnet mask to the subinterface. • Replace *ip-address* with the primary IPv4 address for an interface. • Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways: – The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address. – The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

	Command or Action	Purpose

Step 5	`exit`
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# exit`
		(Optional) Exits the subinterface configuration mode.
		• The **exit** command is not explicitly required.

Step 6	Repeat Step 2 through Step 5 to define the rest of the VLAN subinterfaces.	—

Step 7	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 8	**`show ethernet trunk bundle-ether`** *`instance`*
	Example:
	`RP/0/RSP0/CPU0:router# show ethernet trunk bundle-ether 5`
		(Optional) Displays the interface configuration.
		The Ethernet bundle instance range is from 1 through 65535.

Configuring an Attachment Circuit on a VLAN

Use the following procedure to configure an attachment circuit on a VLAN.

SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE** | **Bundle-Ether**] *interface-path-id.subinterface* **l2transport**
3. **encapsulation dot1q**
4. **l2protocol cpsv** {**tunnel** | **reverse-tunnel**}
5. **end**
   or
   commit

6. **show interfaces** [**GigabitEthernet** | **TenGigE**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure terminal`
		Enters global configuration mode.

Step 2	**`interface`** `[`**`GigabitEthernet`** `|` **`TenGigE`** `|` **`Bundle-Ether`** `|` **`TenGigE`**`]` *`interface-path`*`]` *`id.subinterface`* **`l2transport`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/1/0/0.1 l2transport`
		Enters subinterface configuration and specifies the interface type, location, and subinterface number.
		• Replace the argument with one of the following instances:
		– Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack/slot/module/port*, and a slash between values is required as part of the notation.
		– Ethernet bundle instance. Range is from 1 through 65535.
		• Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.
		• Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation.
		Note You must include the **l2transport** keyword in the command string; otherwise, the configuration creates a Layer 3 subinterface rather that an AC.

Step 3	**`encapsulation dot1q`**
	Example:
	`RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100, untagged`
		Sets the Layer 2 encapsulation of an interface.
		Note The **dot1q vlan** command is replaced by the **encapsulation dot1q** command on the Cisco ASR 9000 Series Router. It is still available for backward-compatibility, but only for Layer 3 interfaces.

Step 4	**`l2protocol cpsv`** `{`**`tunnel`** `|` **`reverse-tunnel`**`}`
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# l2protocol cpsv tunnel`
		Configures Layer 2 protocol tunneling and protocol data unit (PDU) filtering on an Ethernet interface for the following protocols: CDP, PVST+, STP, VTP, where:
		• **tunnel**—Specifies L2PT encapsulation on frames as they enter the interface, and de-encapsulation on frames as they exit they interface.
		• **reverse-tunnel**—Specifies L2PT encapsulation on frames as they exit the interface, and de-encapsulation on frames as they enter the interface.

	Command or Action	Purpose

Step 5	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config-if-l2)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if-l2)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6	**show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id.subinterface* **Example:** `RP/0/RSP0/CPU0:router# show interfaces TenGigE 0/3/0/0.1`	(Optional) Displays statistics for interfaces on the router.

What to Do Next

- To configure a point-to-point pseudowire cross connect on the AC, see the "*Implementing MPLS Layer 2* VPNs" module of the *Cisco ASR 9000 Series Router Multiprotocol Label Switching Configuration Guide.*
- To attach Layer 3 service policies, such as Multiprotocol Label Switching (MPLS) or QoS, to the VLAN, refer to the appropriate Cisco ASR 9000 Series Router configuration guide.

Removing an 802.1Q VLAN Subinterface

This task explains how to remove 802.1Q VLAN subinterfaces that have been previously configured using the "Configuring 802.1Q VLAN Subinterfaces" task in this module.

SUMMARY STEPS

1. **configure**
2. **no interface** {**GigabitEthernet** | **TenGigE** | **Bundle-Ether**] *interface-path-id.subinterface*
3. Repeat Step 2 to remove other VLAN subinterfaces.

4. **end**
   or
   commit
5. **show ethernet trunk bundle-ether** *instance* (Optional)

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`no interface`** `{`**`GigabitEthernet`** `|` **`TenGigE`** `|` **`Bundle-Ether`**`]` *`interface-path-id.subinterface`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# no interface TenGigE 0/2/0/4.10`
		Removes the subinterface, which also automatically deletes all the configuration applied to the subinterface.
		• Replace the *instance* argument with one of the following instances:
		– Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack/slot/module/port*, and a slash between values is required as part of the notation.
		– Ethernet bundle instance. Range is from 1 through 65535.
		• Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.
		Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation.

	Command or Action	Purpose
Step 3	Repeat Step 2 to remove other VLAN subinterfaces.	—

Step 4	**`end`**
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**`show ethernet trunk bundle-ether`** *`instance`*
	Example:
	`RP/0/RSP0/CPU0:router# show ethernet trunk bundle-ether 5`
		(Optional) Displays the interface configuration.
		The Ethernet bundle instance range is from 1 through 65535.

Configuration Examples for VLAN Interfaces

This section contains the following example:

VLAN Subinterfaces: Example, page 645

VLAN Subinterfaces: Example

The following example shows how to create three VLAN subinterfaces at one time:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/2/0/4.1
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 10.0.10.1/24
RP/0/RSP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.2
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 101
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 10.0.20.1/24
RP/0/RSP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.3
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 102
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 10.0.30.1/24
RP/0/RSP0/CPU0:router(config-subif)# commit
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# exit
```

```
RP/0/RSP0/CPU0:router# show ethernet trunk bundle-Ether 1
Trunk                                Sub types         Sub states
Interface      St Ly    MTU    Subs       L2      L3       Up    Down  Ad-Down
BE1            Up L3   1514    1000        0    1000     1000       0        0

Summary                        1000        0    1000     1000       0        0
```

The following example shows how to create two VLAN subinterfaces on an Ethernet bundle:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface bundle-ether 2
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.2.1/24
RP/0/RSP0/CPU0:router(config-if)# exit
RP/0/RSP0/CPU0:router(config)# interface bundle-ether 2.1
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 192.168.100.1/24
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# interface bundle-ether 2.2
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 200
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 192.168.200.1/24
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# commit
```

The following example shows how to create a basic dot1Q AC:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0.1
RP/0/RSP0/CPU0:router(config-subif)# l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-subif)# commit
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# exit
```

The following example shows how to create a Q-in-Q AC:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0.2
RP/0/RSP0/CPU0:router(config-subif)# l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 200 second-dot1q 201
RP/0/RSP0/CPU0:router(config-subif)# commit
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# exit
```

The following example shows how to create a Q-in-Any AC:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0.3
RP/0/RSP0/CPU0:router(config-subif)# l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 300 second-dot1q any
RP/0/RSP0/CPU0:router(config-subif)# commit
RP/0/RSP0/CPU0:router(config-subif)# exit
RP/0/RSP0/CPU0:router(config)# exit
```

Additional References

The following sections provide references related to VLAN interface configuration.

Related Documents

Related Topic	Document Title
Cisco ASR 9000 Series Router master command reference	*Cisco ASR 9000 Series Router Master Commands List*
Cisco ASR 9000 Series Router interface configuration commands	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*
Initial system bootup and configuration information for a Cisco ASR 9000 Series Router using the Cisco IOS XR software.	*Cisco ASR 9000 Series Router Getting Started Guide*
Information about user groups and task IDs	*Cisco ASR 9000 Series Router Interface and Hardware Component Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring the Satellite Network Virtualization (nV) System on the Cisco ASR 9000 Series Router

This module describes the configuration of the Satellite Network Virtualization system on the Cisco ASR 9000 Series Aggregation Services Routers.

Feature History for Configuring Satellite System on Cisco ASR 9000 Series Router

Release	Modification
Release 4.2.1	• Support for Satellite Network Virtualization (Satellite nV) Service was included on the Cisco ASR 9000 Series Router.
Release 4.2.3	• Support for 36-Port 10-Gigabit Ethernet Line Card was included.

Contents

Prerequisites for Configuration

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring the Satellite nV system, you must have these hardware and software installed in your chassis:

- Hardware : Cisco ASR 9000 Series Aggregation Services Routers with ASR 9000 Enhanced Ethernet line cards as the location of Inter Chassis Links and Cisco ASR9000v routers as Satellite boxes.
- Software : Cisco IOS XR Software Release 4.2.1 or later on Cisco ASR 9000 Series Router.

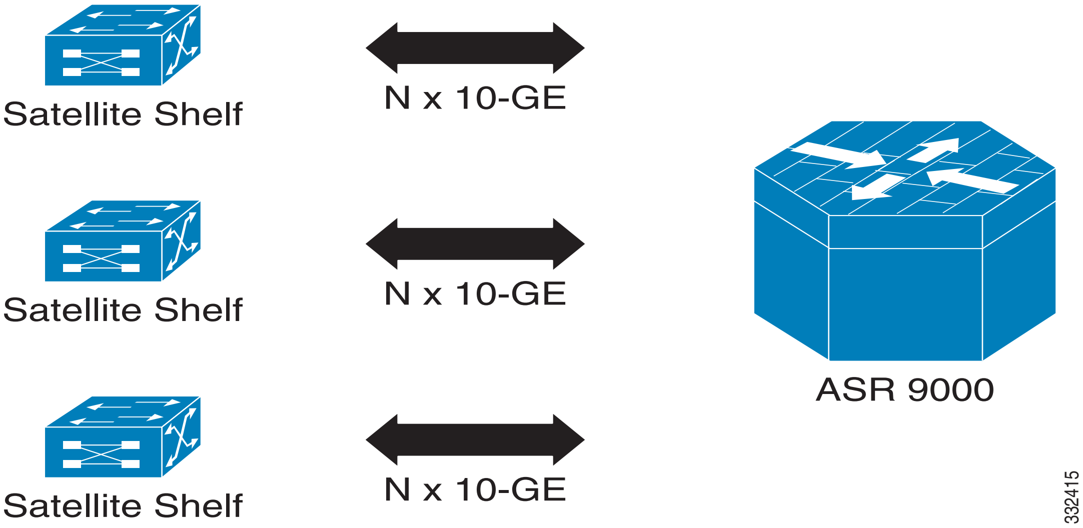
The Satellite nV solution uses the Cisco ASR 9000v as the initial satellite switch. For more information on hardware requirements, see *Cisco ASR 9000 Series Aggregation Services Router Hardware Installation Guide*.

Overview of Satellite nV Switching System

The Cisco ASR 9000 Series Router Satellite Network Virtualization (nV) service or the Satellite Switching System enables you to configure a topology in which one or more satellite switches complement one or more Cisco ASR 9000 Series routers, to collectively realize a single virtual switching system. In this system, the satellite switches act under the management control of the Cisco ASR 9000 Series Aggregation Services Routers. The complete configuration and management of the satellite chassis and features is performed through the control plane and management plane of the Cisco ASR 9000 Series Router.

Interconnection between the Cisco ASR 9000 Series Router and its satellite switches is through standard ethernet interfaces. These are typically 10 Gigabit Ethernet initially but not restricted to any particular flavor or line speed of ethernet. See Figure 1.

Figure 1 Cisco ASR 9000 Series Satellite Switching System



This type of architecture can be realized in a carrier Ethernet transport network, with the satellite switches used as either access switches, or pre-aggregation and aggregation switches. These switches feed into an edge router, such as the Cisco ASR 9000 Series Router or Cisco CRS-3 Router where more advanced L2, L3 services are provisioned.

You can also utilize this model in a FTTB (Fiber To The Business) network application, where business internet and VPN services are offered on a commercial basis. Further, it can also be used in other networks, such as wireless/ RAN backhaul aggregation networks.

Benefits of Satellite nV System

The Cisco ASR 9000 Series satellite nV system offers these benefits:

1. **Extended port scalability and density -** You can create a virtual line card with more than 400 physical Gigabit Ethernet ports. There is a significant increase of ethernet port density in the resulting logical Cisco ASR 9000 Series Router. For example, a single 24-port TenGigE line card on the Cisco ASR 9000 Series Router could integrate up to 24 satellite switches each with 44 GigE ports; this results in an effective port density of 1056 Gigabit Ethernet ports for each Cisco ASR 9000 Series line card slot. In other configurations, even higher port density can be achieved. This is beneficial because the Cisco ASR 9000 Series Router has a per-slot non blocking capacity of up to 400 Gbps (with appropriate RSPs) and there is no other way of physically fitting hundreds of gigabit ethernet ports/ SFPs on the face plate of a single Cisco ASR 9000 Series line card. As a result, in order to utilize the full capacity of an Cisco ASR 9000 Series line card, it is necessary to physically separate out the ethernet ports, while maintaining logical management control. This would appear as if all ports were physically on a single large line card of the Cisco ASR 9000 Series Router.
2. **Reduced cost** - All the edge-routing capabilities and application features of the Cisco IOS XR software are available on low cost access switches.
3. **Reduced operating expense -** You can seamlessly upgrade software images, and also manage the chassis and services from a common point. This includes a single logical router view, single point of applying CLI or XML interface for the entire system of switches, a single point of monitoring the entire system of switches and a single point of image management and software upgrades for the entire system.
4. **Enhanced feature consistency -** All the features on the regular GigE ports of Cisco ASR 9000 Series Router are also available on the access ports of a satellite access switch in a functionally identical and consistent manner. The typical application of a satellite system would be in the access and aggregation layers of a network. By integrating the access switches along with the aggregation or core switch, you can ensure that there are no feature gaps between the access switch and the aggregation or core switch. All features, such as carrier ethernet features, QoS and OAM, function consistently, from access to core, because of this integrated approach.
5. **Improved feature velocity** - With the satellite solution, every feature that is implemented on the Cisco ASR9000 Series Router becomes instantly available at the same time in the access switch, resulting in an ideal feature velocity for the edge switch.
6. **Better resiliency** - The nV satellite solution enables better multi-chassis resiliency, as well as better end-to-end QoS. For more information on QoS capabilities, see *Cisco ASR 9000 Series Aggregation Services Router QoS Configuration Guide*.

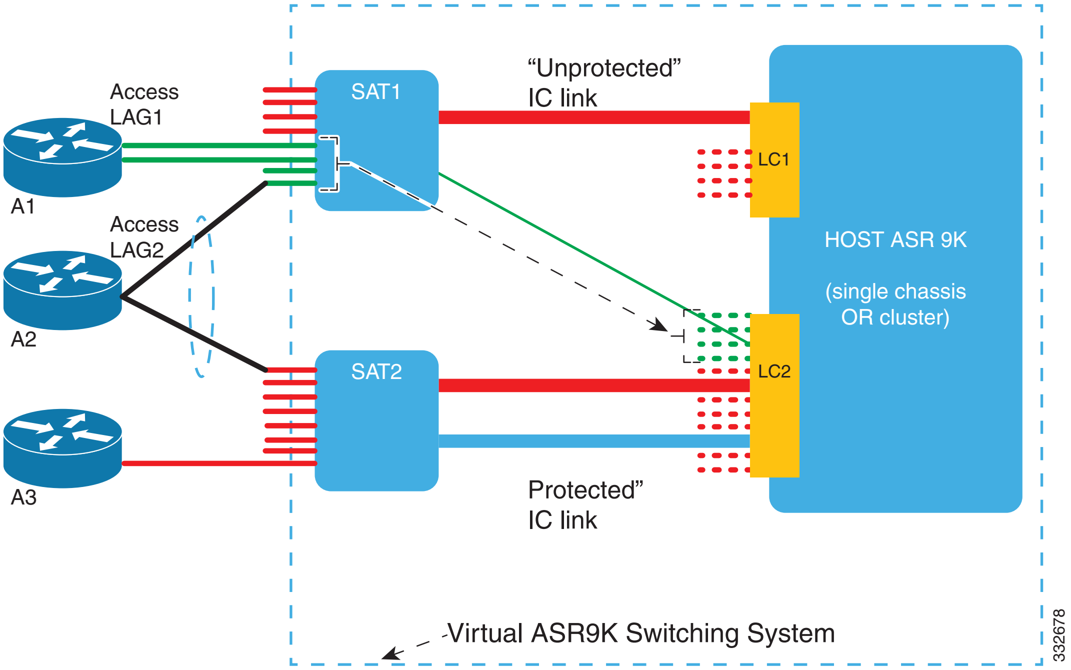
Overview of Port Extender Model

In the Port Extender Satellite switching system, a satellite switch is attached to its parent Cisco ASR 9000 Series Router through physical ethernet ports.

Note In releases later than Cisco IOS XR Software Release 4.2.1, attachment models beyond the port extender model will also be supported.

The parent Cisco ASR 9000 Series Router is referred as the host in this model. From a management or a provisioning point of view, the physical access ports of the satellite switch are equivalent to the physical ethernet ports on the Cisco ASR9000 Series Router. You do not need a specific console connection for managing the Satellite Switching System, except for debugging purposes. The interface and chassis level features of the satellite are visible in the control plane of Cisco IOS XR software running on the host. This allows the complete management of the satellites and the host as a single logical router.

Figure 2 Port Extender Satellite Switching System



In this model, a single Cisco ASR 9000 Series Router hosts two satellite switches, SAT1 and SAT2, to form an overall virtual Cisco ASR 9000 switching system; this is shown by the dotted line surrounding the Cisco ASR 9000 Series Router, SAT1, and SAT2 in Figure 2.

This structure effectively appears as a single logical Cisco ASR 9000 Series Router to the external network. External access switches A1, A2 and A3 connect to this overall virtual switch by physically connecting to SAT1 and SAT2 using normal ethernet links. The links between the satellite switches and the Cisco ASR 9000 Series Router are ethernet links, and are referred as ICLs (Inter-Chassis Links). The Cisco ASR 9000 Series Router is referred as the host in this system. When there is congestion on the interchassis links, an inbuilt QoS protection mechanism is available for the traffic.

Note SAT1, SAT2, and the host Cisco ASR 9000 Series Router need not be located in the same geographic location. This means that the ICLs need not be of nominal length for only intra-location or intra-building use. The ICLs may be tens, hundreds, or even thousands of miles in length, thereby creating a logical satellite switch spanning a large geography.

Note In a Cisco ASR 9000 Series Router multi-chassis cluster system, there are multiple Cisco ASR 9000 Series Router systems within a single virtual switch system. Logically however, it is still considered a single host system.

Features Supported in the Satellite nV System

This section provides details of the features of a satellite system.

Satellite System Physical Topology

The satellite system supports the point-to-point hub and spoke physical topology for the ICLs between satellite switches and the host Cisco ASR 9000 Series Router. This topology allows a physical Ethernet MAC layer connection from the satellite to the Cisco ASR 9000 Series Router. This can be realized using a direct Ethernet over Fiber or Ethernet over Optical transport (such as Ethernet over a SONET/ SDH/ CWDM/ DWDM network).

This topology also allows a satellite switch to be geographically at a separate location, other than that of the host Cisco ASR 9000 Series Router. There is no limit set for the distance, and the solution works even when the satellite is placed at a distance of tens, hundreds, or even thousands of miles from the host.

Inter-Chassis Link Redundancy Modes and Load Balancing

The Cisco ASR 9000 Series Satellite system supports these redundancy modes:

- **Non-redundant inter-chassis links mode** - In this mode, there is no link level redundancy between inter-chassis links of a satellite.
- **Redundant inter-chassis links mode** - In this mode, the link level redundancy between inter-chassis links are provided using a single link aggregation (LAG) bundle.

In the redundant ICL mode, the load balancing of traffic between members of the IC bundle is done using a simple hashing function based on the satellite access port ID, and not based on the flow based hash using L2 or L3 header contents from the packets. This ensures that a single ICL is used by all packets for a given satellite access port. As a result, the actions applied for QoS and other features consider all the packets belonging to a single satellite access port.

For more details on QoS application and configuration on ICLs, see *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide*.

Satellite Discovery and Control Protocols

A Cisco proprietary discovery and control protocol is used between the satellite switches and the host Cisco ASR 9000 Series Router devices, to handle discovery, provisioning, and monitoring of the satellite devices from the host Cisco ASR 9000 Series Satellite System in-band over the ICLs. The Satellite Discovery And Control (SDAC) Protocol provides the behavioural, semantic, and syntactic definition of the relationship between a satellite device and its host.

Satellite Discovery and Control Protocol IP Connectivity

The connectivity for the SDAC protocol is provided through a normal in-band IP routed path over the ICLs using private and public IP addresses appropriate for the carrier's network.

You can configure a management IP address on the host CLI for each satellite switch and corresponding IP addresses on the ICLs. You can select addresses from the private IPv4 address space (for example, 10.0.0.0/8 or 192.1.168.0/24) in order to prevent any conflict with normal service level IPv4 addresses being used in the IPv4 FIB. You can also configure a private VRF that is used for only satellite management traffic, so that the IP addresses assigned to the satellites can be within this private VRF. This reduces the risk of address conflict or IP address management complexity compared to other IP addresses and VRFs that are used on the router.

Layer-2 and L2VPN Features

All L2 and L2VPN features that are supported on physical ethernet or bundle ethernet interfaces are also supported on Satellite Ethernet interfaces. For more details on L2VPN features supported on the Cisco ASR 9000 Series Satellite System, see *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*.

Layer-3 and L3VPN Features

All MPLS L3VPN features that are supported on ethernet interfaces such as GRE, netflow, and so on, are also supported on the Cisco ASR 9000 Series Satellite System. For more information on these features, see *Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide* and *Cisco ASR 9000 Series Aggregation Services Router Netflow Configuration Guide*.

Layer-2 and Layer-3 Multicast Features

All Layer-2 and Layer-3 multicast features, including IGMP, IGMP snooping, PIM, mLDP, MVPN, P2MP TE, are supported on Satellite Ethernet interfaces, as they are supported on normal Ethernet and bundle ethernet interfaces. For more information on these features supported on a satellite system, see *Cisco ASR 9000 Series Aggregation Services Routers Multicast Configuration Guide*.

Quality of Service

Most Layer-2, Layer-3 QoS and ACL features are supported on Satellite Ethernet interfaces that are similar to normal physical Ethernet interfaces, with the exception of any ingress policy with a queueing action. However, for QoS, there may be some functional differences in the behavior because in the Cisco IOS XR Software Release 4.2.1, user-configured MQC policies are applied on the Cisco ASR 9000 Series Router, and not on the satellite switch interfaces. For more detailed information on QoS policy attributes, features, and their configuration, see *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide.*

Note User-configured QoS policies are independent of any default port level QoS that are applied in order to handle IC link congestion and oversubscription scenarios. In addition to the default port-level QoS applied on the satellite system ports, there is also some default QoS applied on the Cisco ASR 9000 Series Router side, to the ingress and egress traffic from and to the Satellite Ethernet ports.

Cluster Support

A cluster of Cisco ASR 9000 Series Routers is supported along with the satellite mode. A single cluster system can act like one logical Cisco ASR 9000 Series Router host system for a group of satellite switches. A satellite switch can also have some ICLs connect to rack 0 and other ICLs connect to rack 1 of a cluster system. For more information, see *Configuring the nV Edge System on the Cisco ASR 9000 Series Router* chapter.

Note The Satellite Ethernet interfaces cannot be used as cluster inter-rack links.

Time of Day Synchronization

The Time of Day parameter on the satellite switch is synchronized with the time of day on the host. This ensures that time stamps on debug messages and other satellite event logs are consistent with the host, and with all satellite switches across the network. This is achieved through the SDAC Discovery Protocol from the host to the satellite switch when the ICLs are discovered.

Satellite Chassis Management

The chassis level management of the satellite is done through the host because the satellite switch is a logical portion of the overall virtual switch. This ensures that service providers get to manage a single logical device with respect to all aspects including service-level, as well as box-level management. This simplifies the network operations. These operations include inventory management, environmental sensor monitoring, and fault/alarm monitoring for the satellite chassis through the corresponding CLI, SNMP, and XML interfaces of the host Cisco ASR 9000 Series Router.

Note The satellite system hardware features, support for SFPs, and compatible topologies are described in the *Cisco ASR 9000 Series Aggregation Services Router Hardware Installation Guide*.

Restrictions of the Satellite nV System

These are some of the software restrictions of the satellite nV system in the current release. These restrictions will be eliminated in the future releases.

- The inter-chassis link redundancy is supported only through the static EtherChannel, and not through LACP based link bundles. Minimum and maximum link commands are not applicable when ICL is a bundle.
- If a satellite system is operating in redundant ICL mode, then you cannot configure link bundles of any form (with or without LACP) on the access ports of that same satellite switch.
- If a satellite system is operating in redundant ICL mode, then Ethernet OAM features are not supported on the access ports of that satellite.
- Multi-chassis Link Aggregation is supported if there are two independent Cisco ASR 9000 Series Routers acting as the POA (Point of Attachment), each with its own satellite switch, and the DHD (Dual Homed Device) connecting through each of the satellite switches. However, MC-LAG is not supported with a single satellite switch that connects two separate Cisco ASR 9000 Series Routers through an ICL LAG.

Note Refer to the *Cisco ASR 9000 Series Aggregation Services Router Release Notes, Release 4.2.1* for additional software restrictions.

Implementing a Satellite nV System

The Interface Control Plane Extender(ICPE) infrastructure has a mechanism to provide the Control Plane of an interface physically located on the Satellite device in the local Cisco IOS XR software. After this infrastructure is established, the interfaces behave like other physical ethernet interfaces on the router.

The ICPE configuration covers these functional areas, which are each required to set up full connectivity with a Satellite device:

- Defining the Satellite nV System, page 714
- Configuring the host IP address, page 717
- Configuring the Inter-Chassis Links and IP Connectivity, page 718
- Plug and Play Satellite nV Switch Turn up: (Rack, Plug, and Go installation), page 721

Defining the Satellite nV System

Each satellite that is to be attached to Cisco IOS XR software must be configured on the host, and also be provided with a unique identifier. In order to provide suitable verification of configuration and functionality, the satellite type, and its capabilities must also be specified.

Further, in order to provide connectivity with the satellite, an IP address must be configured, which will be pushed down to the satellite through the Discovery protocol, and allows Control protocol connectivity.

This task explains how to define the satellite system by assigning an ID and basic identification information.

SUMMARY STEPS

1. **configure**
2. **nv**
3. **satellite** <*Satellite ID*>
4. **serial-number** <*string*> (Optional)
5. **description** <*string*> (Optional)
6. **type** <*type*>
7. **ipv4 address** <*address*>
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**nv** **Example:** `RP/0/RSP0/CPU0:router(config)# nv`	Enters the nV configuration submode.
Step 3	**satellite** *id* **Example:** `RP/0/RSP0/CPU0:router(config-nV)#` `satellite` *<100-65534>*	Declares a new satellite that is to be attached to the host and enters the satellite configuration submode.
Step 4	**serial-number** *<string>* **Example:** `RP/0/RSP0/CPU0:router(config-nV)#` `serial-number` *CAT1521B1BB*	(Optional) Serial number is used for satellite authentication.
Step 5	**description** *id* **Example:** `RP/0/RSP0/CPU0:router(config-nV)#` `description` *Milpitas Building12*	(Optional) Specifies any description string that is associated with a satellite such as location and so on.

Step 6	**type** *type_name*
	Example:
	`RP/0/RSP0/CPU0:router(config-nV)#`
	`satellite 200 type ?`
	`asr9000v  Satellite type`
		Defines the expected type of the attached satellite.

	Command or Action	Purpose
Step 7	**ipv4 address** *address* **Example:** `RP/0/RSP0/CPU0:router(config-nV)# ipv4 address 10.22.1.2`	Specifies the IP address to assign to the satellite. ICPE sets up a connected route to the specified IP address through all configured ICLs.

Step 8	**end**
	or
	commit
	Example:
	`RP/0/0RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the host IP address

This procedure gives you the steps to configure a host IP address on a loopback interface.

SUMMARY STEPS

1. **configure**
2. **interface** *Loopback0*
3. **ipv4 address** 8.8.8.8 255.255.255.255
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	`interface loopback0`
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface loopback0`
		Specifies the loopback address for the interface.

	Command or Action	Purpose

Step 3	`ipv4 address`
	Example:
	`RP/0/RSP0/CPU0:router(config-int)# ipv4 address 8.8.8.8 255.255.255.255`
		Configures the host IP address on a loopback interface.

Step 4	`end`
	or
	`commit`
	Example:
	`RP/0/RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the Inter-Chassis Links and IP Connectivity

Inter-Chassis Links (ICLs) need to be explicitly configured, in order to indicate which satellite is expected to be connected. You must also specify the access, that is down-stream GigE ports, which crosslink up to the Host through the configured ICL. In order to establish connectivity between the host and satellite, suitable IP addresses must be configured on both sides. The satellite IP address is forwarded through the Discovery protocol. The configuration is described in the section, Defining the Satellite nV System, page 714.

Note This configuration shows the use of the global default VRF. The recommended option is to use a private VRF for nV IP addresses as shown in the Satellite Management using private VRF subsection under Satellite System Configuration: Example.

SUMMARY STEPS

1. **configure**
2. **interface** <*interface_name*>
3. **description** *To Sat5 1/46*
4. **ipv4** *point-to-point*
5. **ipv4 unnumbered** *Loopback0*

6. **nv**
7. **satellite-fabric-link satellite** <*id*>
8. **remote-ports** *interface-type*
9. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/RSP0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`interface`** *`interface-name`*
	Example:
	`RP/0/RSP0/CPU0:router(config)# interface TenGigE0/2/1/0`
		The supported inter-chassis link interface types are limited by the connectivity provided on the supported satellites. GigabitEthernet, TenGigE, and Bundle-Ether interfaces are the only support ICL types.

Step 3	**`description`**
	Example:
	`RP/0/RSP0/CPU0:router(config-interface)# description To Sat5 1/46`
		Specifies the description of the supported inter-chassis link interface type.

Step 4	**`ipv4`** *`point-to-point`*
	Example:
	`RP/0/RSP0/CPU0:router(config-interface)# ipv4` *`point-to-point`*
		Configures the IPv4 point to point address.

Step 5	**`ipv4 unnumbered`** *`loopback0`*
	Example:
	`RP/0/RSP0/CPU0:router(config-interface)# interface unnumbered loopback0`
		Configures the IPv4 loopback address on the interface.

Step 6	**`nv`**
	Example:
	`RP/0/RSP0/CPU0:router(config)# nv`
		Enters the nV configuration submode.

	Command or Action	Purpose
Step 7	**satellite-fabric-link satellite** *<id>* **Example:** `RP/0/RSP0/CPU0:router(config-int-nv)# satellite-fabric-link satelite 200`	Specifies that the interface is an ICPE inter-chassis link.
Step 8	**remote-ports** *interface type* **Example:** `RP/0/RSP0/CPU0:router(config-int-nv)# remote-ports GigabitEthernet 0/0/0-30`	Configures the remote satellite ports 0 to 30.

Step 9	**end**
	or
	commit
	Example:
	`RP/0/RSP0/CPU0:router(config)# end`
	or
	`RP/0/RSP0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Note For information on QoS configuration on ICLs , see *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide*.

Configuring the Satellite nV Access Interfaces

The access GigabitEthernet interfaces on the satellite are represented locally in Cisco IOS XR Software using interfaces named GigabitEthernet similar to other non-satellite GigabitEthernet interfaces. The only difference is that the rack id used for a satellite access GigabitEthernet interface is the configured satellite ID for that satellite.

These interfaces support all features that are normally configurable on GigabitEthernet interfaces (if running over a physical IC Link), or Bundle-Ether interfaces (if running over a virtual IC Link).

Plug and Play Satellite nV Switch Turn up: (Rack, Plug, and Go installation)

1. Unpack the Cisco ASR 9000v rack, stack, and connect to the power cord.
2. Plug in Cisco ASR 9000v qualified optics of correct type into any one or more of the SFP+ slots and appropriate qualified optics into SFP+ or XFP slots on the host Cisco ASR 9000 Series Router. Connect through the SMF/MMF fiber.

Note Connect the 10GigE fibers from Cisco ASR 9000 Series Router to any of the 10G SFP+ ports on the Cisco ASR 9000v in any order.

3. Configure the satellite nV system through CLI or XML on the Cisco ASR 9000 Series Router host 10GigE ports. Configure the host for nV operations as described in the sections Defining the Satellite nV System, Configuring the host IP address, and Configuring the Inter-Chassis Links and IP Connectivity.
4. Power up the Cisco ASR 9000v chassis.
5. You can check the status of Cisco ASR 9000v chassis based on these chassis error LEDs on the front face plate.
   - If the Critical Error LED turns ON, then it indicates a serious hardware failure.
   - If the Major Error LED turns ON, then it indicates that the hardware is functioning well but unable to connect to the host.
   - If the Critical and Major LEDs are OFF, then the Cisco ASR 9000v is up and running and connected to the host.
   - You can do satellite ethernet port packet loopback tests through the host, if needed, to check end to end data path.

Note If the satellite software requires an upgrade, it notifies the host Cisco ASR 9000 Series Router. You can do an inband software upgrade from the Cisco ASR 9000 Series Router, if needed. Use the **show nv satellite status** on host Cisco ASR 9000 Series Router to check the status.

Upgrading and Managing Satellite nV Software

Satellite software images are bundled inside a PIE called **asr9k-9000v-nV-p.pie** within the Cisco ASR9000 Series package. The Cisco IOS XR software production SMU tool can be used to generate patches for the satellite image in the field to deliver bug fixes or minor enhancements without requiring a formal software upgrade.

This section provides the commands to manage the satellite nV Software.

Prerequisites

You must have installed the satellite installation procedure using the Plug-and-Play satellite installation procedure. For more information, see Plug and Play Satellite nV Switch Turn up: (Rack, Plug, and Go installation).

Installing a Satellite

To download and activate the software image on the satellite, use the **install nv satellite** <*satellite ID / all*> **transfer/activate** commands**.** The **transfer** command downloads the image to the satellite. When the **activate** command is followed by the transfer command, the software is activated on the satellite.

Example

```
RP/0/RSP0/CPU0:sat-host#install nv satellite 100 transfer
Install operation initiated successfully.
RP/0/RSP0/CPU0:sat-host#RP/0/RSP0/CPU0:May  3 20:12:46.732 : icpe_gco[1146]:
%PKT_INFRA-ICPE_GCO-6-TRANSFER_DONE : Image transfer completed on Satellite 100

RP/0/RSP0/CPU0:sat-host#install nv satellite 100 activate
Install operation initiated successfully.
LC/0/2/CPU0:May  3 20:13:50.363 : ifmgr[201]: %PKT_INFRA-LINK-3-UPDOWN : Interface
GigabitEthernet100/0/0/28, changed state to Down
RP/0/RSP0/CPU0:May  3 20:13:50.811 : invmgr[254]: %PLATFORM-INV-6-OIROUT : OIR: Node 100
removed
```

Note If the **activate** command is run directly, then the software image is transferred to the satellite and also activated.

Example

```
RP/0/RSP0/CPU0:sat-host#install nv satellite 101 activate
Install operation initiated successfully.

RP/0/RSP0/CPU0:sat-host#RP/0/RSP0/CPU0:May  3 20:06:33.276 : icpe_gco[1146]:
%PKT_INFRA-ICPE_GCO-6-TRANSFER_DONE : Image transfer completed on Satellite 101
RP/0/RSP0/CPU0:May  3 20:06:33.449 : icpe_gco[1146]: %PKT_INFRA-ICPE_GCO-6-INSTALL_DONE :
Image install completed on Satellite 101
RP/0/RSP0/CPU0:May  3 20:06:33.510 : invmgr[254]: %PLATFORM-INV-6-OIROUT : OIR: Node 101
removed
```

Note For the satellite image upgrade to work, you must ensure that the management-plane CLI is not configured on the Cisco ASR 9000 Series Router. If it is configured, then you need to add this exception for each of the 10GigE interfaces, which are the satellite ICLs.

You can include the exception using this CLI:

```
control-plane
 management-plane
  inband
    !
   !
   interface TenGigE0/0/0/5  <=== To enable TFTP on nV satellite ICL
allow TFTP
```

If you do not include this exception, then the image download and upgrade fails.

Monitoring the Satellite Software

- To perform a basic status check, use the **show nv satellite status brief** command.

```
RP/0/RSP0/CPU0:shanghai# show nv satellite status brief

Sat-ID  Type      IP Address    MAC address     State
------  --------  ------------  --------------  --------------------------------
100     asr9000v  101.102.103.105  dc7b.9426.1594  Connected (Stable)
200     asr9000v  101.102.103.106  0000.0000.0000  Halted; Conflict: no links configured
400               194.168.9.9   0000.0000.0000  Halted; Conflict: satellite has no type
configured
```

- To check if an upgrade is required on satellite, run the **show nv satellite status satellite** *satellite_id*.

Example

```
RP/0/RSP0/CPU0:sat-host#show nv satellite status satellite 100

Satellite 100
-------------
  State: Connected (Stable)
  Type: asr9000v
  Description: sat-test
  MAC address: dc7b.9427.47e4
  IPv4 address: 100.1.1.1
  Configured Serial Number: CAT1521B1BB
  Received Serial Number: CAT1521B1BB
  Remote version: Compatible (latest version)
    ROMMON: 125.0 (Latest)
    FPGA: 1.13 (Latest)
    IOS: 200.8 (Latest)
  Configured satellite fabric links:
    TenGigE0/2/0/6
    --------------
      State: Satellite Ready
      Port range: GigabitEthernet0/0/0-9
    TenGigE0/2/0/13
    ---------------
      State: Satellite Ready
      Port range: GigabitEthernet0/0/30-39
    TenGigE0/2/0/9
    --------------
      State: Satellite Ready
      Port range: GigabitEthernet0/0/10-19
```

Note In this example’s output, **Remote version**, **ROMMON**, **FPGA**, and **IOS** must show the latest version. If it does not, an upgrade is required on the satellite. The version numbers displayed are the installed version on the ASR 90000v. If a version number is displayed, instead of latest key word in the above output, that would correspond to the ASR9000v image bundles in the satellite pie.

Monitoring the Satellite Protocol Status

- To check the status of the satellite discovery protocol, use the **show nv satellite protocol discovery** command.

```
RP/0/RSP0/CPU0:router# show nv satellite protocol discovery brief
Interface       Sat-ID  Status                          Discovered links
--------------  ------  ------------------------------  -----------------------
Te0/1/0/0       100     Satellite Ready                 Te0/1/0/0
Te0/1/0/1       100     Satellite Ready                 Te0/1/0/1
```

(Or)

```
RP/0/RSP0/CPU0:router# show nv satellite protocol discovery interface TenGigE 0/1/0/0

  Satellite ID: 100
  Status: Satellite Ready
  Remote ports: GigabitEthernet0/0/0-15
  Host IPv4 Address: 101.102.103.104
  Satellite IPv4 Address: 101.102.103.105
  Vendor: cisco, ASR9000v-DC-E
  Remote ID: 2
  Remote MAC address: dc7b.9426.15c2
  Chassis MAC address: dc7b.9426.1594
```

- To check the status of the satellite control protocol status, use the **show nv satellite protocol control** command.

```
RP/0/RSP0/CPU0:shanghai# sh nv satellite protocol control brief
Sat-ID  IP Address     Protocol state  Channels
------  ------------   --------------  -----------------------------------
    101.102.103.105  Connected     Ctrl, If-Ext L1, If-Ext L2, X-link, Soft Reset,
Inventory, EnvMon, Alarm


RP/0/RSP0/CPU0:shanghai# sh nv satellite protocol control
Satellite 100
-------------
  IP address: 101.102.103.105
  Status: Connected
  Channels:
    Control
    -------
      Channel status: Open
      Messages sent: 24 (24 control), received: 23 (23 control).
    Interface Extension Layer 1
    ---------------------------
```

```
      Channel status: Open
      Messages sent: 7 (3 control), received: 14 (2 control).
    Interface Extension Layer 2
    ---------------------------
      Channel status: Open
      Messages sent: 11 (3 control), received: 10 (2 control).
    Interface Extension Cross-link
    ------------------------------
      Channel status: Open
      Messages sent: 4 (3 control), received: 3 (2 control).
….
```

Monitoring the Satellite Inventory

You can use the **show inventory chassis**, **show inventory fans, show environment temperatures** commands in the admin configuration mode to monitor the status of satellite inventory.

```
RP/0/RSP0/CPU0:shanghai(admin)# show inventory chassis

NAME: "module 0/RSP0/CPU0", DESCR: "ASR9K Fabric, Controller, 4G memory"
PID: A9K-RSP-4G, VID: V02, SN: FOC143781GJ
...
NAME: "fantray SAT100/FT0/SP", DESCR: "ASR9000v"
PID: ASR-9000v-FTA, VID: V00 , SN: CAT1507B228

NAME: "module SAT100/0/CPU0", DESCR: "ASR-9000v GE-SFP Line Card"
PID: ASR-9000v, VID: N/A, SN:
NAME: "module mau GigabitEthernet100/0/CPU0/8", DESCR: "CISCO-AVAGO     "
PID: SFP-GE-S, VID: V01, SN: AGM1424P08N

NAME: "module mau TenGigE100/0/CPU0/3", DESCR: "CISCO-FINISAR   "
PID: SFP-10G-SR, VID: V02, SN: FNS144502Y3

NAME: "power-module SAT100/PM0/SP", DESCR: "ASR-9000v Power Module"
PID: ASR-9000v, VID: N/A, SN:
NAME: "Satellite Chassis ASR-9000v ID 100", DESCR: "ASR9000v"
PID: ASR-9000v-AC-A, VID: V00 , SN: CAT12345678
```

```
RP/0/RSP0/CPU0:sat-host (admin)# show inventory fans

NAME: "fantray 0/FT0/SP", DESCR: "ASR-9006 Fan Tray"
PID: ASR-9006-FAN, VID: V02, SN: FOX1519XHU8

NAME: "fantray 0/FT1/SP", DESCR: "ASR-9006 Fan Tray"
PID: ASR-9006-FAN, VID: V02, SN: FOX1519XHTM

NAME: "fantray SAT100/FT0/SP", DESCR: "ASR9000v"
PID: ASR-9000v-FTA, VID: V01 , SN: CAT1531B4TC

NAME: "fantray SAT101/FT0/SP", DESCR: "ASR9000v"
PID: ASR-9000v-FTA, VID: V01 , SN: CAT1542B0LJ

NAME: "fantray SAT102/FT0/SP", DESCR: "ASR9000v"
PID: ASR-9000v-FTA, VID: V01 , SN: CAT1531B4T7
```

```
RP/0/RSP0/CPU0:sat-host(admin)# show inventory | b GigabitEthernet100/

NAME: "module mau GigabitEthernet100/0/CPU0/0", DESCR: "CISCO-FINISAR   "
PID: SFP-GE-S, VID:  , SN: FNS11350L5E

NAME: "module mau GigabitEthernet100/0/CPU0/1", DESCR: "CISCO-FINISAR   "
PID: SFP-GE-S, VID: V01, SN: FNS0934M290

NAME: "module mau GigabitEthernet100/0/CPU0/2", DESCR: "CISCO-FINISAR   "
PID: SFP-GE-S, VID:  , SN: FNS12280L59
```

```
RP/0/RSP0/CPU0:sat-host(admin)# show environment temperatures

R/S/I   Modules Sensor                  (deg C)
0/RSP0/*
        host    Inlet0                  33.1
        host    Hotspot0                46.9

0/RSP1/*
        host    Inlet0                  32.1
        host    Hotspot0                45.9

0/0/*
        host    Inlet0                  37.3
        host    Hotspot0                52.3

0/1/*
        spa0    InletTemp               34.0
        spa0    Hotspot                 34.5

        spa1    LocalTemp               38.0
        spa1    Chan1Temp               36.0
        spa1    Chan2Temp               39.0
        spa1    Chan3Temp               39.0
        spa1    Chan4Temp               48.0

        host    Inlet0                  36.1
        host    Hotspot0                64.0

0/2/*
        host    Inlet0                  39.2
        host    Hotspot0                54.6

0/3/*
        host    Inlet0                  41.3
        host    Hotspot0                48.5

0/FT0/*
        host    Inlet0                  42.3
        host    Hotspot0                36.1

0/FT1/*
        host    Inlet0                  40.4
        host    Hotspot0                35.8

SAT100/FT0/*
        host    Hotspot0                53.0
```

```
SAT101/FT0/*
        host    Hotspot0                56.0


SAT102/FT0/*
        host    Hotspot0                53.0
```

Reloading the Satellite Device

In order to reload the satellite device, use the **hw-module satellite** *satellite id/all* **reload** command.

Example

```
RP/0/RSP0/CPU0:sat-host# hw-module satellite 101 reload

Reload operation completed successfully.
RP/0/RSP0/CPU0:May  3 20:26:51.883 : invmgr[254]: %PLATFORM-INV-6-OIROUT : OIR: Node 101
removed
```

Port Level Parameters Configured on a Satellite

These are the port-level parameters that can be configured on a satellite nV system:

- Admin state (shut and no shut)
- Ethernet MTU
- Ethernet link auto-negotiation that includes,
  - Half and full duplex
  - Link speed
  - Flow control
- Static configuration of auto-negotiation parameters such as speed, duplex, and flow control
- Carrier delay
- Layer-1 packet loopback which includes,
  - Line loopback
  - Internal loopback
- All satellite access port features on Cisco ASR 9000 Series Router.

Configuration Examples for Satellite nV System

This section contains these examples:

Satellite System Configuration: Example

This example shows a sample configuration for setting up the connectivity of a Satellite System.

Satellite Global Configuration

The satellite ID, type, serial number, description, and satellite IP address are configured in the satellite global configuration submode:

```
nv
 satellite 100
  type asr9000v
  serial-number CAT1521B1BB
  description milpitas bldg20
  ipv4 address 10.0.0.100
 !
!
```

ICL (satellite-fabric-link) Interface Configuration

On the interface connected to the satellite (TenGig or Bundle interface), the ports associated with the satellite ID must be specified. All fabric links connected to the same satellite must use the same (host) IPv4 address. The same or different host IPv4 addresses can be used for the same host to connect to different satellites.

```
interface Loopback1000
 ipv4 address 10.0.0.1 255.0.0.0
interface TenGigE0/2/1/0
 description To Sat5 1/46
 ipv4 point-to-point
 ipv4 unnumbered Loopback1000
 nv
  satellite-fabric-link satellite 200
   remote-ports GigabitEthernet 0/0/0-30
  !
 !
!
```

Note These examples illustrate using IP addresses from the global VRF of the router for satellite management traffic. As discussed Satellite Discovery and Control Protocol IP Connectivity section, this can also be done using a private VRF, to prevent IP address conflict with the global VRF. In this case, the loopback interface and the ICL interfaces in the examples must be assigned to the private VRF dedicated for satellite management traffic.

Satellite Interface Configuration

The Satellite interface can be used as any other regular GigabitEthernet interfaces:

```
interface GigabitEthernet200/0/0/0
l2transport
!
!

interface GigabitEthernet200/0/0/0
ip address 99.0.0.1 255.255.255.0
!
!

interface GigabitEthernet200/0/0/2
bundle id 100 mode active
!
!
```

Satellite Management using private VRF

You can use a special private VRF instead of the global default routing table, to configure the loopback interface and ICLs used for satellite management traffic. IP addresses in this VRF will not conflict with any other addresses used on the router.

```
router(config)# vrf NV_MGMT_VRF
router(config)# address ipv4 unicast

router(config)# interface Loopback 1000
router(config)# vrf NV_MGMT_VRF
router(config)# ipv4 address 10.0.0.1 / 24

router(config)# interface TenGige 0/1/0/3
router(config)# vrf NV_MGMT_VRF
router(config)# ipv4 point-to-point
router(config)# ipv4 unnumbered Loopback 1000
router(config)# nv
router(config-nv)# satellite-fabric-link satellite 500
router(config-nv)# remote-ports GigabitEthernet 0/0/28-39
router(config)# nv satellite 500
router(config)# ipv4 address 10.0.0.2 / 24
```

Additional References

These sections provide references to related documents.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Satellite System software upgrade and downgrade on Cisco IOS XR Software	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
Cisco IOS XR interface configuration commands	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*
Satellite QoS configuration information for the Cisco IOS XR software	*Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide*
Layer-2 and L2VPN features on the satellite system	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*
Layer-3 and L3VPN features on the satellite system	*Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide*
Multicast features on the satellite system	*Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide*
Broadband Network Gateway features on the satellite system	*Cisco ASR 9000 Series Aggregation Services Router Broadband Network Gateway Configuration Guide*
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
None	N.A

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring the nV Edge System on the Cisco ASR 9000 Series Router

This module describes the configuration of the nV Edge system on the Cisco ASR 9000 Series Aggregation Services Routers.

Feature History for Configuring nV Edge System on Cisco ASR 9000 Series Router

Release	Modification
Release 4.2.1	• Support for nV Edge system was included on the Cisco ASR 9000 Series Router.

Contents

Prerequisites for Configuration

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring the nV Edge system, you must have these hardware and software installed in your chassis:

- Hardware : Cisco ASR 9000 Series SPA Interface Processor-700 and Cisco ASR 9000 Enhanced Ethernet line cards are supported. Cisco ASR 9000 Enhanced Ethernet line card 10 GigE links are used as IRLs (inter-rack links).
- Software : Cisco IOS XR Software Release 4.2.1 or later on Cisco ASR 9000 Series Router.

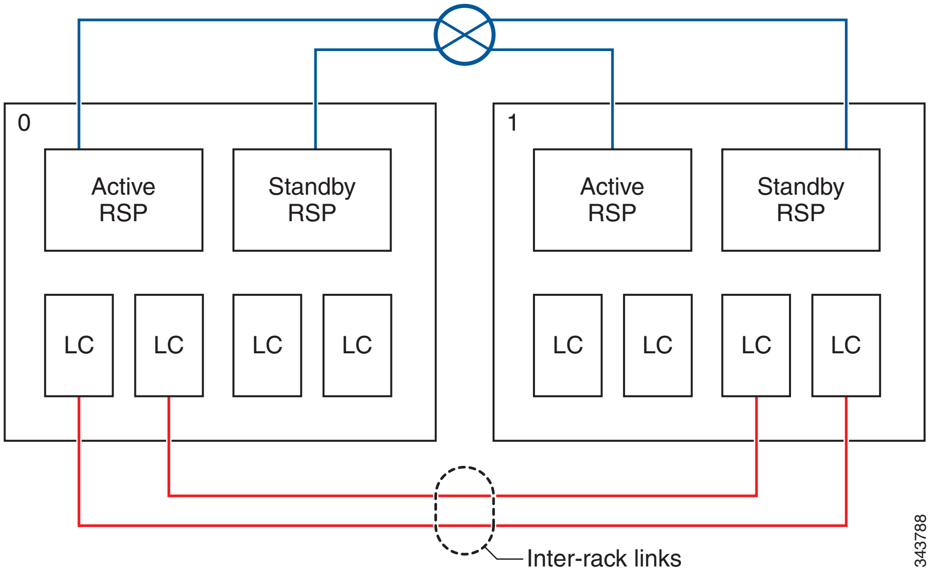
For more information on hardware requirements, see *Cisco ASR 9000 Series Aggregation Services Router Hardware Installation Guide*.

Overview of Cisco ASR 9000 nV Edge Architecture

A Cisco ASR 9000 Series nV Edge consists of two or more Cisco ASR 9000 Series Router chassis that are combined to form a single logical switching or routing entity. You can operate two Cisco ASR 9000 Series Router platforms as a single virtual Cisco ASR 9000 Series system. Effectively, they can logically link two physical chassis with a shared control plane, as if the chassis were two route switch processors (RSPs) within a single chassis. See Figure 40. The blue lines on top shows the internal eobc interconnection and the red lines at the bottom show the data plane interconnection.

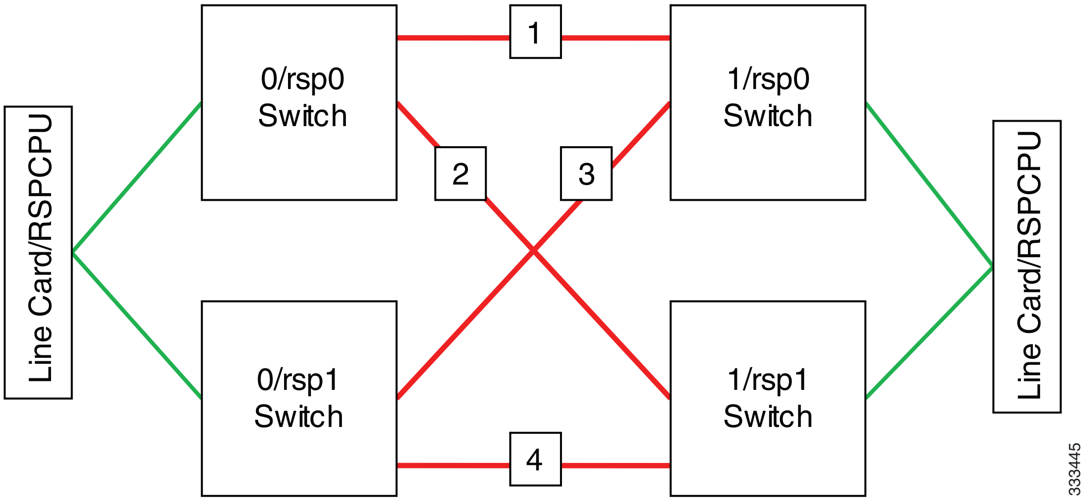
As a result, you can double the bandwidth capacity of single nodes and eliminate the need for complex protocol-based high-availability schemes. Hence, you can achieve failover times of less than 50 milliseconds for even the most demanding services and scalability needs.

Figure 40 Cisco ASR 9000 nV Edge Architecture



Note In Cisco IOS XR Software Release 4.2.1, the scalability of a cluster is limited to two chassis.

Figure 41 EOBC Links on a Cisco ASR 9000 nV Edge System



As illustrated in the Figure 41, the two physical chasses are linked using a Layer 1 1-Gbps connection, with RSPs communicating using a Layer 1 or Layer 2 Ethernet out-of-band channel (EOBC) extension to create a single virtual control plane. Each RSP has 2 EOBC ports and with redundant RSPs there will be 4 connections between the chassis.

The Cisco Virtualized Network Architecture combines the nV Edge system with the satellite devices to offer the Satellite nV architecture. For more information on Satellite nV models, see *Configuring the Satellite Network Virtualization (nV) System on the Cisco ASR 9000 Series Router* chapter.

Inter Rack Links on Cisco ASR 9000 Series nV Edge System

The IRL (Inter Rack Link) connections are required for forwarded traffic going from one chassis out of interface on the other chassis part of the nV edge system. The IRL has to be a 10 GigE link and it has to be direct L1 connections. The IRLs are used for forwarding packets whose ingress and egress interfaces are on separate racks. There can be a maximum of 16 such links between the chassis. A minimum of two links are required and they should be on two separate line cards, for better resiliency in case one line card goes down due to any fault. See Cisco ASR 9000 nV Edge Architecture.

Note For more information on QoS on IRLs, see *Cisco ASR 9000 Series Aggregation Services Router Modular QoS Configuration Guide*.

Failure Detection in Cisco ASR 9000 Series nV Edge System

In the Cisco ASR 9000 Series nV Edge system, when the Primary DSC node fails, the RSP in the Backup DSC node becomes Primary. It executes the duties of the master RSP that hosts the active set of control plane processes. In a normal scenario of nV Edge System where the Primary and Backup DSC nodes are hosted on separate racks, the failure detection for the Primary DSC happens through communication between the racks.

These mechanisms are used to detect RSP failures across rack boundaries:

- FPGA state information detected by the peer RSP in the same chassis is broadcast over the control links. This information is sent if any state change occurs and periodically every 200ms.
- The UDLD state of the inter rack control or data links are sent to the remote rack, with failures detected at an interval of 500ms.
- A keep-alive message is sent between RSP cards through the inter rack control links, with a failure detection time of 10 seconds.

A Split Brain is a condition where the inter rack links between the routers in a Cisco ASR 9000 Series nV Edge system fails and hence the nodes on both routers start to act as primary node. So, messages are sent between these racks in order to detect Split Brain avoidance. These occur at 200ms intervals across the inter-rack data links.

Scenarios for High Availability

These are some sample scenarios for failure detection:

1. Single RSP Failure in the Primary DSC node - The Standby RSP within the same chassis initially detects the failure through the backplane FPGA. In the event of a failure detection, this RSP transitions to the active state and notifies the Backup DSC node about the failure through the inter-chassis control link messaging.
2. Failure of Primary DSC node and the Standby peer RSP - There are multiple cases where this scenario can occur, such as power-cycle of the Primary DSC rack or simultaneous soft reset of both RSP cards within the Primary rack.
   a. The remote rack failure is initially detected by UDLD failure on the inter rack control link. The Backup DSC node checks the UDLD state on the inter rack data link. If the rack failure is confirmed by failure of the data link as well, then the Backup DSC node becomes active.
   b. UDLD failure detection occurs every 500ms but the time between control link and data link failure can vary since these are independent failures detected by the RSP and line cards. A windowing period of up to 2 seconds is needed to correlate the control and data link failures and to allow split brain detection messages to be received. The keep-alive messaging between RSPs acts as a redundant detection mechanism, if the UDLD detection fails to detect a reset RSP card.
3. Failure of Inter Rack Control links (Split Brain) - This failure is initially detected by the UDLD protocol on the Inter Rack Control links. In this case, the Backup DSC continues to receive UDLD and keep-alive messages through the inter rack data link. As discussed in the Scenario 2, a windowing period of two seconds is allowed to synchronize between the control and data link failures. If the data link has not failed, or Split Brain packets are received across the Management LAN, then the Backup DSC rack reloads to avoid the split brain condition.

Benefits of Cisco ASR 9000 Series nV Edge System

The Cisco ASR 9000 Series nV Edge system architecture offers these benefits:

1. The Cisco ASR 9000 Series nV Edge System appears as a single switch or router to the neighboring devices.
2. You can logically link two physical chassis with a shared control plane, as if the chassis were two route switch processors (RSPs) within a single chassis. As a result, you can double the bandwidth capacity of single nodes and eliminate the need for complex protocol-based high-availability schemes.
3. You can achieve failover times of less than 50 milliseconds for even the most demanding services and scalability needs.
4. You can manage the cluster as a single entity rather than two entities. Better resiliency is available due to chassis protecting one another.
5. Cisco nV technology allows you to extend Cisco ASR 9000 Series Router system capabilities beyond the physical chassis with remote virtual line cards. These small form-factor (SFF) Cisco ASR 9000v cards can aggregate hundreds of Gigabit Ethernet connections at the access and aggregation layers.
6. You can scale up to thousands of Gigabit Ethernet interfaces without having to separately provision hundreds or thousands of access platforms. This helps you to simplify the network architecture and reduce the operating expenses (OpEx).
7. The multi-chassis capabilities of Cisco IOS XR Software are employed. These capabilities are extended to allow for enhanced chassis resiliency including data plane, control plane, and management plane protection in case of complete failure of any chassis in the Cisco ASR 9000 Series nV Edge System.
8. You can reduce the number of pseudo wires required for achieving pseudowire redundancy.
9. The nV Edge system allows seamless addition of new chassis. There would be no disruption in traffic or control session flap when a chassis is added to the system.

Restrictions of the Cisco ASR 9000 Series nV Edge System

These are some of the restrictions for the Cisco ASR 9000 nV Edge system:

- The first generation Cisco ASR 9000 Ethernet linecards are not supported.
- Chassis types that are not similar cannot be connected to form an nV edge system.
- SFP-GE-S= is the only Cisco supported SFP that is allowed for all inter rack connections.
- TenGigE SFPs are not supported on EOBC ports.
- The nV Edge control plane links have to be direct physical connections and no network or intermediate routing or switching devices are allowed in between.
- The nV Edge system does not support mixed speed links.
- The nV Edge system does not support the ISM or CGN blade.

Implementing a Cisco ASR 9000 Series nV Edge System

This section explains the implementation of Cisco ASR 9000 Series nV Edge System.

- Configuring Cisco ASR 9000 nV Edge System, page 738

Configuring Cisco ASR 9000 nV Edge System

To bring up the Cisco ASR 9000 nV Cluster, you need to perform these steps outlined in the following subsection.

Single Chassis to Cluster Migration

Consider that there are two individual chassis running Cisco IOS XR Software Release 4.2.x image. Let us refer them as rack0 and rack1 in these steps. If they are already running Cisco IOS XR Software Release 4.2.1 or later, you can avoid the first two steps.

1. You must turbo boot each chassis independently with the Cisco IOS XR Software Release 4.2.1**.**
2. Upgrade the field programmable devices (FPDs). This step is required because Cisco ASR 9000 Series nV Edge requires at least the RSP rommons to be corresponding to the Cisco IOS XR Software Release 4.2.1.
3. **Collect information**: You need to know the chassis serial number for each rack that is to be added to the cluster. On an operating system, you can get this from **show inventory chassis** command. On a system at rommon, you can get the serial number from **bpcookie**.
4. In order to setup the admin configuration on rack0, enter:

```
(admin config) # nv edge control serial <rack 0 serial> rack 0
(admin config) # nv edge control serial <rack 1 serial> rack 1
(admin config) # commit
```

5. Reload Rack 0.
6. Boot the RSPs (if you have two) in Rack 1 into the ROMMON mode. Change the ROMMON variables using these commands:

unset CLUSTER_RACK_ID
unset CLUSTER_NO_BOOT
unset BOOT
sync

7. Power down Rack 1.
8. Physically connect the routers. Connect the inter chassis control links on the front panel of the RSP cards (labelled SFP+ 0 and SFP+ 1) together. Rack0-RSP0 connects to Rack1-RSP0, and similarly for RSP1. You can verify the connections once Rack 1 is up using the **show nv edge control interface** *loc 0/RSP0/CPU0* command.

```
RP/0/RSP0/CPU0:ios# show nv edge control switch interface  loc 0/RSP0/CPU0

Priority lPort             Remote_lPort      UDLD STP
======== =====             ============      ==== ========
0        0/RSP0/CPU0/12    1/RSP0/CPU0/12    UP   Forwarding
1        0/RSP0/CPU0/13    1/RSP1/CPU0/13    UP   Blocking
2        0/RSP1/CPU0/12    1/RSP1/CPU0/12    UP   On Partner RSP
3        0/RSP1/CPU0/13    1/RSP0/CPU0/13    UP   On Partner RSP
```

Note You do not need any explicit command for inter-chassis control links and it is on by default.

9. Bring up Rack 1.
10. You must also connect your Interchassis Data links. You must configure it to be interchassis data link interface using the **nv edge interface** configuration command under the 10Gig interface (only 10Gig) . Ensure that this configuration on both sides of the inter chassis data link (on rack0 and rack1).

    If Bundle-ether is used as the interface, then

    – You must include **lacp system mac h.h.h** in the global configuration mode.

    – You must configure **mac-addr h.h.h** on the Bundle-ether interface.

Note Static MAC on bundle is necessary whether or not the Bundle Ethernet members are sent from the same chassis or a different one.

Note You can verify the Interchassis Data Link operation using the **show nv edge data forwarding** command.

11. After Rack0 and Rack1 comes up fully with all the RSPs and line cards in **XR-RUN** state, the **show dsc** and **show redundancy summary** commands must have similar command outputs as shown in nV Edge System Configuration: Example section.

Configuration Examples for nV Edge System

This section contains the following examples:

- nV Edge System Configuration: Example, page 740

nV Edge System Configuration: Example

The following example shows a sample configuration for setting up the connectivity of a Cisco ASR 9000 Series nV Edge System.

IRL (inter-rack-link) Interface Configuration

```
interfacetenGigE 0/1/1/1
nv
edge
interface
!
```

Cisco nV Edge IRL link Support from 10Gig interface

In this case, te0/2/0/0 and te1/2/0/0 provide Inter Rack datalink:

```
RP/0/RSP0/CPU0:cluster_router#show runn interface te1/2/0/0

interface TenGigE1 /2/0/0
nv
edge
data
interface
!

RP/0/RSP0/CPU0:cluster_router#show runn interface te0/2/0/0

interface TenGigE0 /2/0/0
nv
edge
data
interface
!
```

Additional References

The following sections provide references to related documents.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*
Satellite System software upgrade and downgrade on Cisco IOS XR Software	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
Cisco IOS XR interface configuration commands	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*
Satellite Qos configuration information for the Cisco IOS XR software	*Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Configuration Guide*
Layer-2 and L2VPN features on the Satellite system	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide*
Layer-3 and L3VPN features on the Satellite system	*Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide*
Multicast features on the Satellite system	*Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide*
Broadband Network Gateway features on the Satellite system	*Cisco ASR 9000 Series Aggregation Services Router Broadband Network Gateway Configuration Guide*
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	Description
CISCO-RF-MIB	Provides DSC chassis active/standby node pair information. In nV Edge scenario, it provides DSC primary/backup RP information and switchover notification. To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml
ENTITY-STATE-MIB	Provides redundancy state information for each node.
CISCO-ENTITY-STATE-EXT-MIB	Extension to ENTITY-STATE-MIB which defines notifications (traps) on redundancy status changes.
CISCO-ENTITY-REDUNDANCY-MIB	Defines redundancy group types such as Node Redundancy group type and Process Redundancy group type.

RFCs

RFCs	Title
None	N.A

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

INDEX

HC	Cisco IOS XR Interface and Hardware Component Configuration Guide
IC	Cisco IOS XR IP Addresses and Services Configuration Guide
MCC	Cisco IOS XR Multicast Configuration Guide
MNC	Cisco IOS XR System Monitoring Configuration Guide
MPC	Cisco IOS XR MPLS Configuration Guide
QC	Cisco IOS XR Modular Quality of Service Configuration Guide
RC	Cisco IOS XR Routing Configuration Guide
SBC	Cisco IOS XR Session Border Controller Configuration Guide
SC	Cisco IOS XR System Security Configuration Guide
SMC	Cisco IOS XR System Management Configuration Guide
VFC	Cisco IOS XR Virtual Firewall Configuration Guide