



L2VPN and Ethernet Services Configuration Guide for Cisco NCS 5000 Series Routers, IOS XR Release 6.3.x

First Published: 2017-09-01

Last Modified: 2018-03-01

Americas Headquarters

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

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: <https://www.cisco.com/c/en/us/about/legal/trademarks.html>. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2017–2018 Cisco Systems, Inc. All rights reserved.

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

Cisco Bug Search Tool

[Cisco Bug Search Tool](#) (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.

© 2017–2018 Cisco Systems, Inc. All rights reserved.



CONTENTS

PREFACE

Preface ix

Changes to This Document **ix**

Obtaining Documentation and Submitting a Service Request **ix**

CHAPTER 1

New and Changed VPN Features 1

New and Changed VPN Features **1**

CHAPTER 2

Configure Gigabit Ethernet for Layer 2 VPNs 3

Introduction to Layer 2 Virtual Private Networks **3**

Introduction to Layer 2 VPNs on Gigabit Ethernet Interfaces **4**

Configure Gigabit Ethernet Interfaces for Layer 2 Transport **5**

Running Configuration **5**

Verification **6**

Configure Link Loss Forwarding for Layer 2 Transport **6**

Ethernet Data Plane Loopback **6**

Configure Ethernet Data Plane Loopback **8**

Running Configuration **9**

Verification **10**

Related Topics **11**

Associated Commands **11**

Ethernet Local Management Interface (E-LMI) **11**

E-LMI Messaging **12**

E-LMI Operation **12**

Configure Ethernet Local Management Interface (E-LMI) **13**

Running Configuration **14**

Verify the Ethernet Local Management Interface (E-LMI) Configuration **15**

CHAPTER 3	Configure Virtual LANs in Layer 2 VPNs	19
	Configure VLAN Sub-Interfaces	20
	Introduction to Ethernet Flow Point	22
	Identify Frames of an EFP	23
	Apply Features	23
	Define Data-Forwarding Behavior	24
	Configure VLAN Header Rewrite	24
	Valid Ingress Rewrite Actions	27
	Valid Ingress-Egress Rewrite Combinations	28

CHAPTER 4	Configure Link Bundles for Layer 2 VPNs	33
	Configure Gigabit Ethernet Link Bundle	33
	Configure VLAN Bundle	36
	References for Configuring Link Bundles	37
	Characteristics of Link Bundles	38
	Methods of Forming Bundles of Ethernet Interfaces	38
	Link Aggregation Through LACP	39

CHAPTER 5	Configure Multipoint Layer 2 Services	41
	Prerequisites for Implementing Multipoint Layer 2 Services	41
	Information About Implementing Multipoint Layer 2 Services	42
	Multipoint Layer 2 Services Overview	42
	Bridge Domain	42
	Pseudowires	42
	Access Pseudowire is not supported over VPLS Bridge Domain	42
	Virtual Forwarding Instance	43
	VPLS for an MPLS-based Provider Core	43
	VPLS for Layer 2 Switching	43
	Interoperability Between Cisco IOS XR and Cisco IOS on VPLS LDP Signaling	44
	Pseudowire Redundancy	44
	Configuration	44
	MAC Address-related Parameters	47
	MAC Address Flooding	47

MAC Address-based Forwarding	47
MAC Address Source-based Learning	47
MAC Address Aging	47
MAC Address Limit	48
MAC Address Withdrawal	48
Configuration Examples for Multipoint Layer 2 Services	48
Multipoint Layer 2 Services Configuration for Provider Edge-to-Provider Edge: Example	49
Multipoint Layer 2 Services Configuration for Provider Edge-to-Customer Edge: Example	49
Displaying MAC Address Withdrawal Fields: Example	50
Bridging on IOS XR Trunk Interfaces: Example	52
Bridging on Ethernet Flow Points: Example	56

CHAPTER 6**Configure L2VPN Autodiscovery and Signaling 61**

L2VPN Autodiscovery and Signaling	61
BGP-based VPLS Autodiscovery	61
BGP-based VPLS Autodiscovery with BGP Signaling	61
Configuring BGP and LDP for BGP-based Autodiscovery	62
Configuring BGP-based VPLS Autodiscovery with BGP Signaling	63
BGP-based VPLS Autodiscovery with LDP Signaling	64
Configuring BGP-based VPLS Autodiscovery with LDP Signaling	65
BGP-based VPWS Autodiscovery	66
BGP-based VPWS Autodiscovery with BGP Signaling	66
Configuring BGP-based VPWS Autodiscovery with BGP Signaling	66
BGP-based VPWS Autodiscovery with LDP Signaling	72
Configuring BGP-based VPWS Autodiscovery with LDP Signaling	72

CHAPTER 7**Storm Control 75**

Storm Control	75
Supported Traffic Types for Storm Control	76
Storm Control Thresholds	76
Restrictions	76
Configure Storm Control	76
Related Topics	77
Associated Commands	77

CHAPTER 8	Configure Multiple Spanning Tree Protocol	79
	Overview of Spanning Tree Protocol	79
	Restrictions for STP on Cisco NCS 5000 Series Routers	79
	Overview of MSTP	80
	MSTP Support on Cisco NCS 5000 Series Routers	80
	MSTP BPDU Guard	80
	Flush Containment	81
	Bringup Delay	81
	Configuring MSTP	82
	Running Configuration for MSTP	83
	Verification for MSTP	84
	Configuring MSTP BPDU Guard	84
	Running Configuration with MSTP BPDU Guard	85
	Verification for MSTP BPDU Guard	85
	References for Spanning Tree Protocol	86
	STP Operation	86
	Topology Changes	86
	Variants of STP	87

CHAPTER 9	References	89
	Gigabit Ethernet Protocol Standards	89
	Carrier Ethernet Model References	89
	Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet	91
	References for Configuring Link Bundles	92
	Characteristics of Link Bundles	92
	Methods of Forming Bundles of Ethernet Interfaces	93
	Link Aggregation Through LACP	93



Preface

This product has reached end-of-life status. For more information, see the [End-of-Life and End-of-Sale Notices](#).

This preface contains these sections:

- [Changes to This Document, on page ix](#)
- [Obtaining Documentation and Submitting a Service Request, on page ix](#)

Changes to This Document

This table lists the technical changes made to this document since it was first released.

Table 1: Changes to This Document

Date	Summary
September 2017	Initial release of this document.
March 2018	Republished for Release 6.3.2.

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.



CHAPTER 1

New and Changed VPN Features

This table summarizes the new and changed feature information for the L2VPN and Ethernet Services Configuration Guide for Cisco NCS 5000 Series Routers, and tells you where they are documented.

- [New and Changed VPN Features, on page 1](#)

New and Changed VPN Features

Table 2: VPN Features Added or Modified in IOS XR Release 6.3.x

Feature	Description	Changed in Release	Where Documented
Traffic Storm Control	Traffic Storm Control provides Layer 2 port security under a Virtual Private LAN Services (VPLS) bridge by preventing excess traffic from disrupting the bridge.	Release 6.3.1	Storm Control, on page 75
Multiple Spanning Tree Protocol	The Multiple Spanning Tree Protocol (MSTP) is a Spanning Tree Protocol (STP) variant that allows multiple and independent spanning trees to be created over the same physical network.	Release 6.3.1	Configure Multiple Spanning Tree Protocol, on page 79
L2VPN Autodiscovery and Signaling	L2VPN Autodiscovery and Signaling enables the discovery of remote Provider Edge (PE) routers and the associated signaling in order to provision the pseudowires	Release 6.3.1	Configure L2VPN Autodiscovery and Signaling, on page 61



CHAPTER 2

Configure Gigabit Ethernet for Layer 2 VPNs

This chapter introduces you to Layer 2 features and standards, and describes how you can configure L2VPN features.

The distributed Gigabit Ethernet (including 10-Gigabit and 100-Gigabit) 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.

- [Introduction to Layer 2 Virtual Private Networks, on page 3](#)
- [Introduction to Layer 2 VPNs on Gigabit Ethernet Interfaces, on page 4](#)
- [Configure Gigabit Ethernet Interfaces for Layer 2 Transport, on page 5](#)
- [Configure Link Loss Forwarding for Layer 2 Transport, on page 6](#)
- [Ethernet Data Plane Loopback, on page 6](#)
- [Ethernet Local Management Interface \(E-LMI\), on page 11](#)
- [E-LMI Messaging, on page 12](#)
- [E-LMI Operation, on page 12](#)
- [Configure Ethernet Local Management Interface \(E-LMI\) , on page 13](#)

Introduction to Layer 2 Virtual Private Networks

A Layer 2 Virtual Private Network (VPN) emulates a physical sub-network in an IP or MPLS network, by creating private connections between two points. Building a L2VPN network requires coordination between the service provider and customer. The service provider establishes Layer 2 connectivity. The customer builds a network by using the data link resources obtained from the service provider. In a L2VPN service, the service provider does not require information about the customer's network topology and other information. This helps maintain customer privacy, while using the service provider resources to establish the network.

The service provider requires Provider Edge (PE) routers with the following capabilities:

- Encapsulation of L2 protocol data units (PDU) into Layer 3 (L3) packets.
- Interconnection of any-to-any L2 transports.
- Support for MPLS tunneling mechanism.
- Process databases that include all information related to circuits and their connections.

This section introduces Layer 2 Virtual Private Networks (VPNs) and the corresponding Gigabit Ethernet services.

Introduction to Layer 2 VPNs on Gigabit Ethernet Interfaces

A L2VPN network enables service providers (SPs) to provide L2 services to geographically disparate customer sites. Typically, a SP uses an access network to connect the customer to the core network. This access network may use a mixture of L2 technologies, such as Ethernet and Frame Relay. The connection between the customer site and the nearby SP edge router is known as an attachment circuit (AC). Traffic from the customer travels over this link to the edge of the SP core network. The traffic then tunnels through a pseudowire over the SP core network to another edge router. The edge router sends the traffic down another AC to the customer's remote site.

The L2VPN feature enables the connection between different types of L2 attachment circuits and pseudowires, allowing users to implement different types of end-to-end services.



Note BOOTP traffic (dst UDP 68) over any type of pseudowire is unsupported.

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 pseudowire, regardless of any VLAN tags that are present on the packets. In Port 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). To configure L2VPN on VLANs, see *The Carrier Ethernet Model* chapter in this manual. In VLAN mode, the configuration is performed under the individual sub-interface.

Switching can take place in the following ways:

- **AC-to-PW**—Traffic reaching the PE is tunneled over a PW (pseudowire) (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.

**Note**

- 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.
- The **encapsulation dot1ad *vlan-id*** and **encapsulation dot1ad *vlan-id* dot1q any** commands cannot co-exist on the same physical interface or bundle interface. Similarly, the **encapsulation dot1q *vlan-id*** and **encap dot1q *vlan-id* second-dot1q any** commands cannot co-exist on the same physical interface or bundle interface. If there is a need to co-exist, it is recommended to use the exact keyword in the single tag encapsulation. For example, **encap dot1ad *vlan-id* exact** or **encap dot1q *vlan-id* exact**.
- In an interface which already has QinQ configuration, you cannot configure the QinQ Range sub-interface where outer VLAN range of QinQ Range overlaps with outer VLAN of QinQ. Attempting this configuration results in the splitting of the existing QinQ and QinQ Range interfaces. However, the system can be recovered by deleting a recently configured QinQ Range interface.
- In an interface which already has QinQ Range configuration, you cannot configure the QinQ Range sub-interface where outer VLAN range of QinQ Range overlaps with inner VLAN of QinQ Range. Attempting this configuration results in the splitting of the existing QinQ and QinQ Range interfaces. However, the system can be recovered by deleting a recently configured QinQ Range interface.

You can use the **show interfaces** command to display AC and pseudowire information.

Configure Gigabit Ethernet Interfaces for Layer 2 Transport

This section describes how you can configure Gigabit ethernet interfaces for Layer 2 transport.

```

/* Enter the interface configuration mode */
Router# configure
Router(config)# interface TenGigE 0/0/0/10

/* Configure the ethertype for the 802.1q encapsulation (optional) */
/* For VLANs, the default ethertype is 0x8100. In this example, we configure a value of
0x9100.
/* The other assignable value is 0x9200 */
/* When ethertype is configured on a physical interface, it is applied to all sub-interfaces
created on this interface */

Router(config-if)# dot1q tunneling ethertype 0x9100

/* Configure Layer 2 transport on the interface, and commit your configuration */
Router(config-if)# l2transport
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# commit

```

Running Configuration

```

configure
interface TenGigE 0/0/0/10
dot1q tunneling ethertype 0x9100
l2transport

```

!

Verification

Verify that the 10-Gigabit Ethernet interface is up and operational.

```
router# show interfaces TenGigE 0/0/0/10
...
TenGigE0/0/0/10 is up, line protocol is up
  Interface state transitions: 1
  Hardware is TenGigE, address is 0011.1aac.a05a (bia 0011.1aac.a05a)
  Layer 1 Transport Mode is LAN
  Layer 2 Transport Mode
  MTU 1514 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
    reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation ARPA,
  Full-duplex, 10000Mb/s, link type is force-up
  output flow control is off, input flow control is off
  Carrier delay (up) is 10 msec
  loopback not set,
  ...
```

Configure Link Loss Forwarding for Layer 2 Transport

Link Loss Forwarding (LLF) is supported on Cisco router. The LLF is used to avoid any packet loss and trigger the network convergence through alternate links.

LLF sends signals across the PW to the neighbouring device to bring the PW and far-end AC down if the local AC goes down. The LLF feature supports the **l2transport propagate remote-status** command used to propagate Layer 2 transport events.

LLF is supported for TenGigE and GigE interfaces and not supported on the Bundle interfaces.

Running Configuration

```
/* Configuring propagation remote-status */
interface TenGigE 0/0/0/5
  l2transport
    propagate remote-status
  !
  !
```

Ethernet Data Plane Loopback

The Ethernet Data Plane Loopback function allows you to run loopback tests to test the connectivity and quality of connections through a Layer 2 cloud. You can run this test on:

- main interface or sub-interfaces
- bundle or its sub-interfaces
- multiple hops through the underlying network

You can use this feature to test the throughput of an ethernet port remotely. You can verify the maximum rate of frame transmission with no frame loss.

This feature allows for bidirectional or unidirectional throughput measurement, and on-demand or out-of-service (intrusive) operation during service turn-up.

Two types of Ethernet loopback are supported:

- External loopback - Traffic loopback occurs at the Ingress interface. Traffic does not flow into the router for loopback.
- Internal loopback - Traffic loopback occurs at the Egress interface. Traffic loopback occurs after the traffic flows into the router to the other interface.

Ethernet data traffic can be looped back on per port basis. This feature supports a maximum of 100 concurrent Ethernet data plane loopback sessions per system. Filters based on frame header can be used for initiating the loopback session. This ensures that only a subset of traffic that is received on an interface is looped back. You can use Source MAC, Destination MAC, and VLAN Priority (COS bits) as filters.

Ethernet Data Plane Loopback Configuration Restrictions

These configuration restrictions are applicable for Ethernet Data Plane Loopback:

- Ethernet data plane loopback is not supported on L3 interfaces or L3 sub-interfaces.
- The following filters are not supported:
 - Outer VLAN or range of outer VLAN
 - Inner VLAN or range of inner VLAN
 - Ether type
- Only the following combination of filters are supported for external loopback:
 - Source MAC
 - Source MAC and Destination MAC
 - Source MAC, Destination MAC, and VLAN priority
 - Destination MAC
 - Destination MAC and VLAN priority
- The re-write modification on the loopback traffic is not supported.
- Ethernet data plane loopback is not supported on BVI interface.
- Only one Ethernet loopback session, either internal or external, can be active on the same interface at any given instance.
- This feature supports a maximum throughput of 10Gbps for internal loopback over all the sessions. For external loopback, there is no throughput limit.
- Dropping of packets received in the non-loopback direction is not supported.
- Ethernet data plane loopback is not supported on packets having destination as multicast MAC address.
- External and internal Ethernet data plane loopback is not supported over bridge domain.

Configure Ethernet Data Plane Loopback

This section describes how you can configure Ethernet Data Plane Loopback on physical interface and sub-interface. Configuring Ethernet Data Plane Loopback involves these steps:

- Configuring Ethernet Data Plane External Loopback
- Starting an Ethernet Data Plane Loopback Session

Configuration Example

```

/* Configuring Ethernet Data Plane External Loopback */

/* On physical interface */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface tenGigE 0/0/0/0 l2transport
RP/0/RSP0/CPU0:router((config-if-l2)# ethernet loopback permit external

/* Starting an Ethernet Data Plane Loopback Session */

RP/0/RSP0/CPU0:router# ethernet loopback start local interface tenGigE 0/0/0/0 external
source mac-address 0000.0000.0001 destination mac-address 0000.0000.0002 cos 5 timeout none

/* On physical sub-interface */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface tenGigE 0/2/0/0/0.1 l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router((config-if-l2)# ethernet loopback permit external

/* Starting an Ethernet Data Plane Loopback Session */

RP/0/RSP0/CPU0:router# ethernet loopback start local interface tenGigE 0/2/0/0/0.1 external
source mac-address 0000.0000.0001 destination mac-address 0000.0000.0002 cos 5 timeout
none

/* Configuring Ethernet Data Plane Internal Loopback */

/* On physical interface */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface tenGigE 0/0/0/1 l2transport
RP/0/RSP0/CPU0:router((config-if-l2)# ethernet loopback permit internal

/* Starting an Ethernet Data Plane Loopback Session */

RP/0/RSP0/CPU0:router# ethernet loopback start local interface tenGigE 0/0/0/1 internal
source mac-address 0000.0000.0002 destination mac-address 0000.0000.0003 cos 5 timeout none

/* On physical sub-interface */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface tenGigE 0/2/0/0/0.1 l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-if-l2)# ethernet loopback permit internal

/* Starting an Ethernet Data Plane Loopback Session */

```

```
RP/0/RSP0/CPU0:router# ethernet loopback start local interface tenGigE 0/2/0/0/0.1 internal
source mac-address 0000.0000.0002 destination mac-address 0000.0000.0003 cos 5 timeout
none
```

```
/* Stopping an Ethernet Data Plane Loopback Session */
```

```
RP/0/RSP0/CPU0:router# ethernet loopback stop local interface tenGigE 0/0/0/0 id 1
RP/0/RSP0/CPU0:router# ethernet loopback stop local interface tenGigE 0/0/0/1 id 2
RP/0/RSP0/CPU0:router# ethernet loopback stop local interface tenGigE 0/2/0/0/0.1 id 1
```

Similarly, you can configure the Ethernet Data Plane Loopback session for bundle interface and bundle sub-interface.

Running Configuration

This section shows Ethernet Data Plane Loopback running configuration.

```
/* External Loopback */

/* On physical interface */

configure
interface interface tenGigE 0/0/0/0 l2transport
  ethernet loopback permit external
!

/* On physical sub-interface */

configure
interface interface tenGigE 0/2/0/0/0.1 l2transport
  encapsulation dot1q 100
  ethernet loopback permit external
!

/* Internal Loopback */

/* On physical interface */

configure
interface interface tenGigE 0/0/0/1 l2transport
  ethernet loopback permit internal
!

/* On physical sub-interface */

configure
interface interface tenGigE 0/2/0/0/0.1 l2transport
  encapsulation dot1q 100
  ethernet loopback permit internal
!
```

Verification

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

```
RP/0/RSP0/CPU0:router# show ethernet loopback permitted
```

```
-----
Interface                               Dot1q(s)                               Direction
-----
tenGigE 0/0/0/1.1                       100                                     Internal
tenGigE 0/0/0/0.1                       100                                     External
-----
```

```
/* This example shows all active sessions on the router */
```

```
RP/0/RSP0/CPU0:router# show ethernet loopback active
```

```
Thu Jul 20 11:00:57.864 UTC
Local: TenGigE0/0/0/0.1, ID 1
=====
Direction:                               External
Time out:                                 None
Time left:                                 -
Status:                                    Active
Filters:
  Dot1Q:                                   Any
  Second-dot1Q:                            Any
  Source MAC Address:                       Any
  Destination MAC Address:                  Any
  Class of Service:                         Any
Local: TenGigE0/0/0/0.1, ID 2
=====
Direction:                               External
Time out:                                 None
Time left:                                 -
Status:                                    Active
Filters:
  Dot1Q:                                   Any
  Second-dot1Q:                            Any
  Source MAC Address:                       0000.0000.0001
  Destination MAC Address:                  0000.0000.0002
  Class of Service:                         5
```

Related Topics

- [Ethernet Data Plane Loopback, on page 6](#)

Associated Commands

- ethernet loopback
- show ethernet loopback

Related Topics

- [Ethernet Data Plane Loopback](#), on page 6

Associated Commands

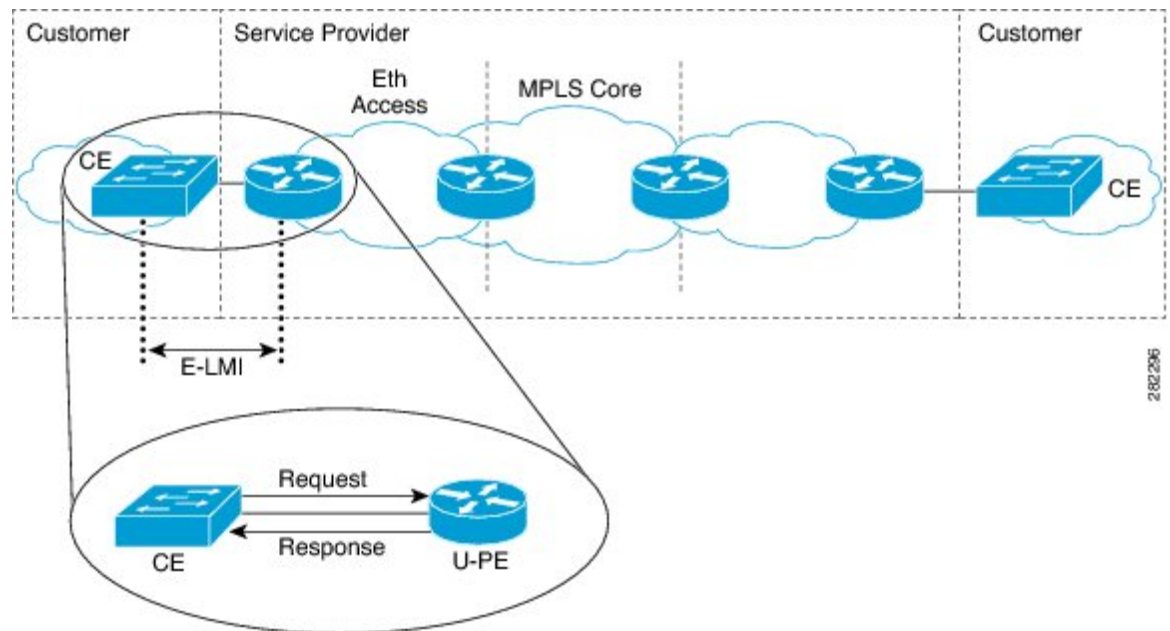
- ethernet loopback
- show ethernet loopback

Ethernet Local Management Interface (E-LMI)

The Cisco NCS 5500 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 this figure).

Figure 1: 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.

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.

Configure Ethernet Local Management Interface (E-LMI)

Before you configure E-LMI on the 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.

E-LMI is not supported on physical sub-interfaces and bundle main and sub- interfaces. E-LMI is configurable on Ethernet physical interfaces only.

In order to ensure the correct interaction between the CE and the PE, each device has two configurable parameters. The CE uses a Polling Timer (PT) and a Polling Counter; the PE uses a Polling Verification Timer (PVT) and a Status Counter.

To configure Ethernet LMI, complete the following tasks:

- Configure EVCs for E-LMI (required)
- Configure Ethernet CFM for E-LMI (required)
- Enable E-LMI on the Physical Interface (required)
- Configure the Polling Verification Timer (optional)
- Configure the Status Counter (optional)

```

/* Configure EVCs for E-LMI/

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/3/0/9/1.1 l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 1
RP/0/RSP0/CPU0:router(config-subif)# xconnect group evpn
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group evpn
RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p p1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface TenGigE0/3/0/9/1.1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor evpn evi 1 target 3001 source 1
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)#commit

/* Configure Ethernet CFM for E-LMI */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)#interface TenGigE0/3/0/9/1.1 l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 1
RP/0/RSP0/CPU0:router(config-subif)# ethernet cfm
RP/0/RSP0/CPU0:router(config-if-cfm)# mep domain irf_evpn_up service up_mep_evpn_1 mep-id
3001
RP/0/RSP0/CPU0:router(config-if-cfm-mep)#exit
RP/0/RSP0/CPU0:router(config)#ethernet cfm
RP/0/RSP0/CPU0:router(config-cfm)# domain irf_evpn_up level 3 id null
RP/0/RSP0/CPU0:router(config-cfm-dmn)#service up_mep_evpn_1 xconnect group evpn p2p p1 id
number 1
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# mip auto-create all ccm-learning
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)# continuity-check interval 1m loss-threshold 3
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#continuity-check archive hold-time 10
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#mep crosscheck

```

```
RP/0/RSP0/CPU0:router(config-cfm-xcheck)# mep-id 1
RP/0/RSP0/CPU0:router(config-cfm-xcheck)#ais transmission interval 1m cos 6
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#log ais
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#log continuity-check errors
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#log crosscheck errors
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#log continuity-check mep changes
RP/0/RSP0/CPU0:router(config-cfm-dmn-svc)#commit
```

```
/* Enable E-LMI on the Physical Interface */
```

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)#interface TenGigE0/3/0/9/1
RP/0/RSP0/CPU0:router(config-if)# ethernet lmi
RP/0/RSP0/CPU0:router(config-if-elmi)#commit
```

```
/* Configure 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.

```
RP/0/RSP0/CPU0:router# configure
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-elmi)#polling-verification-timer 30
RP/0/RSP0/CPU0:router(config-if-elmi)#commit
```

```
/* Configure 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.

```
RP/0/RSP0/CPU0:router# configure
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-elmi)#status-counter 5
RP/0/RSP0/CPU0:router(config-if-elmi)#commit
```

Running Configuration

This section shows E-LMI running configuration.

```
/* Configure EVCs for E-LMI */

configure
interface TenGigE0/3/0/9/1.1 l2transport
 encapsulation dot1q 1

!

l2vpn
xconnect group evpn
 p2p p1
 interface TenGigE0/3/0/9/1.1
 neighbor evpn evi 1 target 3001 source 1
```



```

    commit
!

/* Configure Ethernet CFM for E-LMI */

configure
interface TenGigE0/3/0/9/1.1 l2transport
encapsulation dot1q 1
ethernet cfm
  mep domain irf_evpn_up service up_mep_evpn_1 mep-id 3001
!
configure
ethernet cfm
  domain irf_evpn_up level 3 id null
  service up_mep_evpn_1 xconnect group evpn p2p p1 id number 1
  mip auto-create all ccm-learning
  continuity-check interval 1m loss-threshold 3
  continuity-check archive hold-time 10
  mep crosscheck
  mep-id 1
  !
  ais transmission interval 1m cos 6
  log ais
  log continuity-check errors
  log crosscheck errors
  log continuity-check mep changes
!

/* Enable E-LMI on the Physical Interface */

configure
interface TenGigE0/3/0/9/1
ethernet lmi
!

/* Configure the Polling Verification Timer */

configure
interface gigabitethernet 0/0/0/0
ethernet lmi
  polling-verification-timer 30
!

/* Configure the Status Counter */

configure
interface gigabitethernet 0/0/0/0
ethernet lmi
  status-counter 5
!

```

Verify the Ethernet Local Management Interface (E-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: TenGigE0/3/0/9/1
Ether LMI Link Status: Up
Line Protocol State: Up
MTU: 1514 (1 PDU reqd. for full report)
CE-VLAN/EVC Map Type: Service Multiplexing with no bundling (1 EVC)
Configuration: Status counter 4, Polling Verification Timer 15 seconds
Last Data Instance Sent: 130
Last Sequence Numbers: Sent 179, Received 108

Reliability Errors:
  Status Enq Timeouts          0 Invalid Sequence Number      0
  Invalid Report Type          0

Protocol Errors:
  Malformed PDUs              0 Invalid Protocol 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 00:03:17 ago  Full Status Sent      00:03:17 ago
PDU Received            00:00:07 ago  PDU Sent              00:00:07 ago
LMI Link Status Changed 01:59:54 ago  Last Protocol Error   never
Counters Cleared        never

Sub-interface: TenGigE0/3/0/9/1.1
VLANs: 1
EVC Status: Active
EVC Type: Point-to-Point
OAM Protocol: CFM
  CFM Domain: irf_evpn_up (level 3)
  CFM Service: up_mep_evpn_1

Remote UNI Count: Configured = 1, Active = 1
  Remote UNI Id
  -----
  <Remote UNI Reference Id: 1>
                                     Status
                                     -----
                                     Up

```

Make sure:

- The protocol (Ether LMI Link Status) is 'Up'.
- The output does not have "local UNI (UNI Id)" and also it is in provisioned state.
- The interface (Line Protocol State) is 'Up'.
- The CE-VLAN/EVC Map Type is as expected and shows the correct number of EVCs.
- The error counters are all 0.
- The LMI Link Status Changed timer shows the time since the protocol started.
- The sub-interface name(s) corresponds to the EFP(s) configured.
- The VLANs on each interface are as configured.
- The EVC Status is 'Active'.
- The CFM Domain and CFM Service match the provisioning.

- The Remote UNI Id is as provisioned.

Verify CFM (UP 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)
* - Multiple errors received        S - Standby

Domain irf_evpn_up (level 3), Service up_mep_evpn_1
Up MEP on TenGigE0/3/0/9/1.1 MEP-ID 3001
=====
St   ID MAC Address   Port   Up/Downtime   CcmRcvd SeqErr   RDI Error
-----
>   1 008a.964b.6410 Up     00:09:59           12    0    0    0
=====
```

Ensure St is >, which means it is OK(up)

Related Topics

- [Ethernet Local Management Interface \(E-LMI\), on page 11](#)
- [E-LMI Messaging, on page 12](#)
- [E-LMI Messaging, on page 12](#)

Associated Commands

- ethernet lmi
- show ethernet lmi interfaces
- show ethernet cfm peer meps



CHAPTER 3

Configure Virtual LANs in Layer 2 VPNs

The Layer 2 Virtual Private Network (L2VPN) feature enables Service Providers (SPs) to provide L2 services to geographically disparate customer sites.

A virtual local area network (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. The IEEE's 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames.

VLANs are very useful for user and host management, bandwidth allocation, and resource optimization. Using VLANs addresses the problem of breaking large networks into smaller parts so that broadcast and multicast traffic does not consume more bandwidth than necessary. VLANs also 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 sub-interface configuration on Gigabit Ethernet and 10-Gigabit Ethernet interfaces.

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 sub-interface, and then configures that VLAN in sub-interface configuration mode. To create an Attachment Circuit, you need to include the **l2transport** keyword in the **interface** command string to specify that the interface is a L2 interface.

VLAN ACs support the following modes of L2VPN operation:

- Basic Dot1Q Attachment Circuit—The Attachment Circuit covers all frames that are received and sent with a specific VLAN tag.
- QinQ Attachment Circuit—The Attachment Circuit 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.

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).

Restrictions and Limitations

To configure VLANs for Layer 2 VPNs, the following restrictions are applicable.

- In a point-to-point connection, the two Attachment Circuits do not have to be of the same type. For example, a port mode Ethernet Attachment Circuit can be connected to a Dot1Q Ethernet Attachment Circuit.

- Pseudowires can run in VLAN mode or in port mode. A pseudowire running in VLAN mode always carries Dot1Q or Dot1ad tag(s), while a pseudowire running in port mode may or may NOT carry tags. To connect these different types of circuits, popping, pushing, and rewriting tags is required.
- The Attachment Circuits on either side of an MPLS pseudowire can be of different types. In this case, the appropriate conversion is carried out at one or both ends of the Attachment Circuit to pseudowire connection.
- You can program a maximum number of 16 virtual MAC addresses on your router.
- [Configure VLAN Sub-Interfaces, on page 20](#)
- [Introduction to Ethernet Flow Point, on page 22](#)
- [Configure VLAN Header Rewrite, on page 24](#)

Configure VLAN Sub-Interfaces

Sub-interfaces 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.

Sub-interfaces are distinguished from one another by adding an extension on the end of the interface name and designation. For instance, the Ethernet sub-interface 23 on the physical interface designated TenGigE 0/1/0/0 would be indicated by TenGigE 0/1/0/0.23.

Before a sub-interface is allowed to pass traffic, it must have a valid tagging protocol encapsulation and VLAN identifier assigned. All Ethernet sub-interfaces always default to the 802.1Q VLAN encapsulation. However, the VLAN identifier must be explicitly defined.

The sub-interface Maximum Transmission Unit (MTU) is inherited from the physical interface with 4 bytes allowed for the 802.1Q VLAN tag.

The following modes of VLAN sub-interface configuration are supported:

- Basic dot1q Attachment Circuit
- Q-in-Q Attachment Circuit

To configure a basic dot1q Attachment Circuit, use this encapsulation mode:

encapsulation dot1q *vlan extra-id*

To configure a basic dot1ad Attachment Circuit, use this encapsulation mode:

encapsulation dot1ad *vlan-id*

To configure a Q-in-Q Attachment Circuit, use the following encapsulation modes:

- **encapsulation dot1q** *vlan-id* **second-dot1q** *vlan-id*
- **encapsulation dot1ad** *vlan-id* **dot1q** *vlan-id*

Restrictions and Limitations

To configure VLAN sub-interface, the following restrictions are applicable.

- For double tagged packet, the VLAN range is supported only on the inner tag.

- VLAN list is not supported.

VLANs separated by comma are called a VLAN list. See the example below.

```
Router(config)#interface tenGigE 0/0/0/2.0 l2transport
Router(config-subif)#encapsulation dot1q 1,2 >> VLAN range with comma
Router(config-subif)#commit
```

- If 0x9100/0x9200 is configured as tunneling ether-type, then dot1ad (0x88a8) encapsulation is not supported.
- If any sub-interface is already configured under a main interface, modifying the tunneling ether-type is not supported.
- You can program a maximum number of 16 virtual MAC addresses on your router.
- Following limitations are applicable to both outer and inner VLAN ranges:
 - 32 unique VLAN ranges are supported per system.
 - The overlap between outer VLAN ranges on sub-interfaces of the same Network Processor Unit (NPU) is not supported. A sub-interface with a single VLAN tag that falls into a range configured on another sub-interface of the same NPU is also considered an overlap.
 - The overlap between inner VLAN ranges on sub-interfaces of the same NPU is not supported.
 - Range 'any' does not result in explicit programming of a VLAN range in hardware and therefore does not count against the configured ranges.

Configuration Example

Configuring VLAN sub-interface involves:

- Creating a Ten Gigabit Ethernet sub-interface
- Enabling L2 transport mode on the interface
- Defining the matching criteria (encapsulation mode) to be used in order to map ingress frames on an interface to the appropriate service instance

Configuration of Basic dot1q Attachment Circuit

```
Router# configure
Router(config)# interface TenGigE 0/0/0/10.1 l2transport
Router(config-if)# encapsulation dot1q 10
Router(config-if)# no shutdown
```

Running Configuration

```
configure
interface TenGigE 0/0/0/10.1
  l2transport
  encapsulation dot1q 10
!
```

!

Verification

Verify that the VLAN sub-interface is active:

```

router# show interfaces TenGigE 0/0/0/10.1
...
TenGigE0/0/0/10.1 is up, line protocol is up
  Interface state transitions: 1
  Hardware is VLAN sub-interface(s), address is 0011.1aac.a05a
  Layer 2 Transport Mode
  MTU 1518 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
    reliability Unknown, txload Unknown, rxload Unknown
  Encapsulation 802.1Q Virtual LAN,
    Outer Match: Dot1Q VLAN 10
    Ethertype Any, MAC Match src any, dest any
  loopback not set,
...

```

Associated Commands

- [encapsulation dot1ad dot1q](#)
- [encapsulation dot1q](#)
- [encapsulation dot1q second-dot1q](#)
- [l2transport \(Ethernet\)](#)
- [encapsulation dot1ad](#)

Introduction to Ethernet Flow Point

An Ethernet Flow Point (EFP) is a Layer 2 logical sub-interface used to classify traffic under a physical or a bundle interface. An EFP is defined by a set of filters (a set of entries) that are applied to all the ingress traffic to classify the frames that belong to a particular EFP. Each entry usually contains 0, 1 or 2 VLAN tags. You can specify a VLAN or QinQ tagging to match against on ingress. A packet that starts with the same tags as an entry in the filter is said to match the filter; if the start of the packet does not correspond to any entry in the filter, then the packet does not match the filter.

All traffic on ingress are processed by that EFP if a match occurs, and this can in turn change VLAN IDs, add or remove VLAN tags, and change ethertypes. After the frames are matched to a particular EFP, any appropriate feature (such as, any frame manipulations specified by the configuration as well as things such as QoS and ACLs) can be applied.

The benefits of EFP include:

- Identifying all frames that belong to a particular flow on a given interface
- Performing VLAN header rewrites

(See, [Configure VLAN Header Rewrite, on page 24](#))

- Adding features to the identified frames
- Optionally defining how to forward the identified frames in the data path

Limitations of EFP

Egress EFP filtering is not supported on Cisco IOS XR.

Identify Frames of an EFP

The EFP identifies frames belonging to a particular flow on a given port, independent of their Ethernet encapsulation. An EFP can flexibly map frames into a flow or EFP based on the fields in the frame header. The frames can be matched to an EFP using VLAN tag(s).

The frames cannot be matched to an EFP through this:

- Any information outside the outermost Ethernet frame header and its associated tags such as
 - IPv4, IPv6, or MPLS tag header data
 - C-DMAC, C-SMAC, or C-VLAN

VLAN Tag Identification

Below table describes the different encapsulation types and the EFP identifier corresponding to each.

Encapsulation Type	EFP Identifier
Single tagged frames	802.1Q customer-tagged Ethernet frames
Double tagged frames	802.1Q (ethertype 0x8100) double tagged frames 802.1ad (ethertype 0x88a8) double tagged frames

You can use wildcards while defining frames that map to a given EFP. EFPs can distinguish flows based on a single VLAN tag, a stack of VLAN tags or a combination of both (VLAN stack with wildcards). It provides the EFP model, a flexibility of being encapsulation agnostic, and allows it to be extensible as new tagging or tunneling schemes are added.

Apply Features

After the frames are matched to a particular EFP, any appropriate features can be applied. In this context, “features” means any frame manipulations specified by the configuration as well as things such as QoS and ACLs. The Ethernet infrastructure provides an appropriate interface to allow the feature owners to apply their features to an EFP. Hence, IM interface handles are used to represent EFPs, allowing feature owners to manage their features on EFPs in the same way the features are managed on regular interfaces or sub-interfaces.

The only L2 features that can be applied on an EFP that is part of the Ethernet infrastructure are the L2 header encapsulation modifications. The L2 features are described in this section.

Encapsulation Modifications

EFP supports these L2 header encapsulation modifications on both ingress and egress:

- Push 1 or 2 VLAN tags
- Pop 1 or 2 VLAN tags



Note This modification can only pop tags that are matched as part of the EFP.

- Rewrite 1 or 2 VLAN tags:
 - Rewrite outer tag
 - Rewrite outer 2 tags
 - Rewrite outer tag and push an additional tag
 - Rewrite outer tag and pop inner tag

For each of the VLAN ID manipulations, these can be specified:

- The VLAN tag type, that is, C-VLAN, S-VLAN, or I-TAG. The ethertype of the 802.1Q C-VLAN tag is defined by the dot1q tunneling type command.
- The VLAN ID. 0 can be specified for an outer VLAN tag to generate a priority-tagged frame.



Note For tag rewrites, the CoS bits from the previous tag should be preserved in the same way as the DEI bit for 802.1ad encapsulated frames.

Define Data-Forwarding Behavior

The EFP can be used to designate the frames belonging to a particular Ethernet flow forwarded in the data path. These forwarding cases are supported for EFPs in Cisco IOS XR software:

- L2 Switched Service (Bridging)—The EFP is mapped to a bridge domain, where frames are switched based on their destination MAC address. This includes multipoint services:
 - Ethernet to Ethernet Bridging
 - Multipoint Layer 2 Services
- L2 Stitched Service (AC to AC xconnect)—This covers point-to-point L2 associations that are statically established and do not require a MAC address lookup.
 - Ethernet to Ethernet Local Switching—The EFP is mapped to an S-VLAN either on the same port or on another port. The S-VLANs can be identical or different.
- Tunneled Service (xconnect)—The EFP is mapped to a Layer 3 tunnel. This covers point-to-point services, such as EoMPLS.

Configure VLAN Header Rewrite

EFP supports the following VLAN header rewrites on both ingress and egress ports:

- Push 1 VLAN tag

- Pop 1 VLAN tag



Note This rewrite can only pop tags that are matched as part of the EFP.

- Translate 1 or 2 VLAN tags:
 - Translate 1-to-1 tag: Translates the outermost tag to another tag
 - Translate 1-to-2 tags: Translates the outermost tag to two tags
 - Translate 2-to-1 tag: Translates the outermost two tags to a single tag
 - Translate 2-to-2 tags: Translates the outermost two tags to two other tags

Various combinations of ingress, egress VLAN rewrites with corresponding tag actions during ingress and egress VLAN translation, are listed in the following sections:

- [Valid Ingress Rewrite Actions, on page 27](#)
- [Valid Ingress-Egress Rewrite Combinations, on page 28](#)

Configuration Example

This topic covers VLAN header rewrites on various attachment circuits, such as:

- L2 single-tagged sub-interface
- L2 double-tagged sub-interface

Configuring VLAN header rewrite involves:

- Creating a TenGigabit Ethernet sub-interface
- Enabling L2 transport mode on the interface
- Defining the matching criteria (encapsulation mode) to be used in order to map single-tagged frames ingress on an interface to the appropriate service instance
- Specifying the encapsulation adjustment that is to be performed on the ingress frame

Configuration of VLAN Header Rewrite (single-tagged sub-interface)

```
Router# configure
Router(config)# interface TenGigE 0/0/0/10.1 12transport
Router(config-if)# encapsulation dot1q 10
Router(config-if)# rewrite ingress tag push dot1q 20 symmetric
```

Running Configuration

```
/* Configuration without rewrite */

configure
```

```

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
!
!

/* Configuration with rewrite */

/* PUSH 1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
 rewrite ingress tag push dot1q 20 symmetric
!
!

/* POP 1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
 rewrite ingress tag pop 1
!
!

/* TRANSLATE 1-1 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
 rewrite ingress tag translate 1-to-1 dot1q 20
!
!

/* TRANSLATE 1-2 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10
 rewrite ingress tag translate 1-to-2 dot1q 20 second-dot1q 30
!
!

```

Running Configuration (VLAN header rewrite on double-tagged sub-interface)

```

/* Configuration without rewrite */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
!
!

/* Configuration with rewrite */

/* PUSH 1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
 rewrite ingress tag push dot1q 20 symmetric
!
!

/* TRANSLATE 1-1 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
 rewrite ingress tag translate 1-to-1 dot1q 20
!
!

```

```

/* TRANSLATE 1-2 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
  rewrite ingress tag translate 1-to-2 dot1q 20 second-dot1q 30
!
!

/* TRANSLATE 2-1 */
interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
 rewrite ingress tag translate 2-to-1 dot1q 20

/* TRANSLATE 2-2 */

interface TenGigE0/0/0/0.1 l2transport
 encapsulation dot1q 10 second-dot1q 11
  rewrite ingress tag translate 2-to-2 dot1q 20 second-dot1q 30
!
!

```

Associated Commands

- [encapsulation dot1ad dot1q](#)
- [encapsulation dot1q](#)
- [encapsulation dot1q second-dot1q](#)
- [l2transport \(Ethernet\)](#)
- [rewrite ingress tag](#)

Valid Ingress Rewrite Actions

Table 3: Valid Ingress Rewrite Actions

Interface Configuration	Ingress Rewrite Action
dot1q	No rewrite
dot1q	Pop 1
dot1q	Push 1
dot1q	Push 2
dot1q	Translate 1 to 1
dot1q	Translate 1 to 2
QinQ	No rewrite
QinQ	Pop 1
QinQ	Push 1

Interface Configuration	Ingress Rewrite Action
QinQ	Translate 1 to 1
QinQ	Translate 1 to 2
QinQ	Translate 2 to 1
Untagged	No rewrite

The following notations are used for the rewrite actions mentioned in the table:

- Translate 1-to-1 tag: Translates the outermost tag to another tag.
- Translate 1-to-2 tags: Translates the outermost tag to two tags.
- Translate 2-to-1 tags: Translates the outermost two tags to a single tag.
- Translate 2-to-2 tags: Translates the outermost two tags to two other tags.

Valid Ingress-Egress Rewrite Combinations

Table 4: Valid Ingress-Egress Rewrite Combinations

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q	No rewrite	dot1q	No rewrite
dot1q	No rewrite	dot1q	Pop 1
dot1q	No rewrite	dot1q	Push 1
dot1q	No rewrite	dot1q	Translate 1-to-1
dot1q	Pop 1	dot1q	No rewrite
dot1q	Pop 1	dot1q	Pop 1
dot1q	Push 1	dot1q	No rewrite
dot1q	Push 1	dot1q	Push 1
dot1q	Push 1	dot1q	Push 2
dot1q	Push 1	dot1q	Translate 1-to-1
dot1q	Push 1	dot1q	Translate 1-to-2
dot1q	Push 2 / Translate 1-to-2	dot1q	Push 1
dot1q	Push 2 / Translate 1-to-2	dot1q	Push 2
dot1q	Push 2 / Translate 1-to-2	dot1q	Translate 1-to-2

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
dot1q	Translate 1-to-1	dot1q	No rewrite
dot1q	Translate 1-to-1	dot1q	Push 1
dot1q	Translate 1-to-1	dot1q	Translate 1-to-1
dot1q	No rewrite / Translate 1-to-1	QinQ	No rewrite
dot1q	No rewrite / Translate 1-to-1	QinQ	Pop 1
dot1q	No rewrite / Translate 1-to-1	QinQ	Push 1
dot1q	No rewrite / Translate 1-to-1	QinQ	Translate 1-to-1
dot1q	Pop 1	QinQ	No rewrite
dot1q	Pop 1	QinQ	Pop 1
dot1q	Push 1	QinQ	No rewrite
dot1q	Push 1	QinQ	Pop 1
dot1q	Push 1	QinQ	Push 1
dot1q	Push 1	QinQ	Translate 1-to-1
dot1q	Push 1	QinQ	Translate 1-to-2
dot1q	Push 1	QinQ	Translate 2-to-2
dot1q	Push 2 / Translate 1-to-2	QinQ	No rewrite
dot1q	Push 2 / Translate 1-to-2	QinQ	Push 1
dot1q	Push 2 / Translate 1-to-2	QinQ	Translate 1-to-1
dot1q	Push 2 / Translate 1-to-2	QinQ	Translate 1-to-2
dot1q	Push 2 / Translate 1-to-2	QinQ	Translate 2-to-2
dot1q	No rewrite	Untagged	No rewrite
dot1q	Pop 1	Untagged	No rewrite
dot1q	Push 1	Untagged	No rewrite
dot1q	Push 1	Untagged	No rewrite
dot1q	Push 2	Untagged	No rewrite
dot1q	Translate 1-to-1	Untagged	No rewrite
dot1q	Translate 1-to-2	Untagged	No rewrite

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	No rewrite
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Pop 1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Push 1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Translate 1-to-1
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Translate 1-to-2
QinQ	No rewrite / push 1 / Translate 1-to-1	QinQ	Translate 2-to-2
QinQ	Pop 1	QinQ	No rewrite
QinQ	Pop 1	QinQ	Pop 1
QinQ	Pop 1	QinQ	Push 1
QinQ	Pop 1	QinQ	Translate 1-to-1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	No rewrite
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Push 1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Translate 1-to-1
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Translate 1-to-2
QinQ	Translate 1-to-2 / Translate 2-to-2	QinQ	Translate 2-to-2
QinQ	No rewrite	Untagged	No rewrite
QinQ	No rewrite	Untagged	No rewrite
QinQ	No rewrite	Untagged	No rewrite
QinQ	Pop 1	Untagged	No rewrite
QinQ	Pop 1	Untagged	No rewrite
QinQ	Push 1 / Translate 1-to-1	Untagged	No rewrite
QinQ	Push 1 / Translate 1-to-1	Untagged	No rewrite
QinQ	Translate 1-to-2 / Translate 2-to-2	Untagged	No rewrite

Ingress Interface Configuration	Ingress Interface Rewrite Action	Egress Interface Configuration	Egress Interface Rewrite Action
Untagged	No rewrite	Untagged	No rewrite

The following notations are used for the rewrite actions mentioned in the table:

- Translate 1-to-1 tag: Translates the outermost tag to another tag
- Translate 1-to-2 tags: Translates the outermost tag to two tags
- Translate 2-to-2 tags: Translates the outermost two tags to two other tags



Note The following rewrites are not supported for EoMPLS:

- Push 2
 - Pop 2
 - Translate 1-to-2
 - Translate 2-to-1
 - Translate 2-to-2
-



CHAPTER 4

Configure Link Bundles for Layer 2 VPNs

An ethernet 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 or QoS).

The advantages of link bundling are:

- Redundancy - Because bundles have multiple links, the failure of a single link does not cause a loss of connectivity.
- Increased bandwidth - On bundled interfaces traffic is forwarded over all available members of the bundle aggregating individual port capacity.

There are two types of link bundling supported depending on the type of interface forming the bundle:

- Ethernet interfaces
- VLAN interfaces (bundle sub-interfaces)

For more information, see [References for Configuring Link Bundles, on page 37](#).

This section describes the configuration of ethernet and VLAN link bundles for use in Layer 2 VPNs.

- [Configure Gigabit Ethernet Link Bundle, on page 33](#)
- [Configure VLAN Bundle, on page 36](#)
- [References for Configuring Link Bundles, on page 37](#)

Configure Gigabit Ethernet Link Bundle

Cisco IOS XR software supports the EtherChannel method of forming bundles of Ethernet interfaces. EtherChannel is a 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.

IEEE 802.3ad encapsulation employs a Link Aggregation Control Protocol (LACP) to ensure that all the member links in an ethernet bundle are compatible. Links that are incompatible or have failed are automatically removed from the bundle.

Cisco NCS 5000 Series Router supports 10G and 100G link bundles.

Restrictions

- All links within a single ethernet link bundle must be configured either to run 802.3ad (LACP) or Etherchannel (non-LACP). Mixed links within a single bundle are not supported.

- A combination of 10G and 100G links is not supported in the same ethernet link bundle.
- MAC accounting is not supported on Ethernet link bundles.
- The maximum number of supported links in each ethernet link bundle is 32 .
- The maximum number of supported ethernet link bundles is 64 .

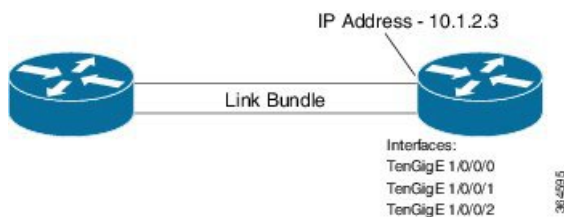
Configuration Example

To create a link bundle between two routers, you must complete the following configurations:

1. Create a bundle instance
2. Map physical interface (s) to the bundle.

Sample values are provided in the following figure.

Figure 2: Link Bundle Topology



For an Ethernet bundle to be active, you must perform the same configuration on both connection endpoints of the bundle.

Configuration

```

/* Enter the global configuration mode and create the ethernet link bundle */
Router# configure
Router(config)# interface Bundle-Ether 3
Router(config-if)# ipv4 address 10.1.2.3 255.0.0.0
Router(config-if)# bundle maximum-active links 32 hot-standby
Router(config-if)# bundle minimum-active links 1
Router(config-if)# bundle minimum-active bandwidth 30000000
Router(config-if)# exit

/* Map physical interfaces to the bundle */
/* Note: Mixed link bundle mode is supported only when active-standby operation is configured
*/
Router(config)# interface TenGigE 1/0/0/0
Router(config-if)# bundle id 3 mode on
Router(config-if)# no shutdown
Router(config)# exit

Router(config)# interface TenGigE 1/0/0/1
Router(config-if)# bundle id 3 mode on
Router(config-if)# no shutdown
Router(config-if)# exit

Router(config)# interface TenGigE 1/0/0/2
Router(config-if)# bundle id 3 mode on
Router(config-if)# no shutdown
Router(config-if)# exit

```

Running Configuration

```
Router# show running-configuration
configure
interface Bundle-Ether 3
  ipv4 address 10.1.2.3 255.0.0.0
  bundle maximum-active links 32 hot-standby
  bundle minimum-active links 1
  bundle minimum-active bandwidth 30000000
!
interface TenGigE 1/0/0/0
  bundle-id 3 mode on
!
interface TenGigE 1/0/0/1
  bundle-id 3 mode on
!
interface TenGigE 1/0/0/2
  bundle-id 3 mode on
!
```

Verification

Verify that interfaces forming the bundle are active and the status of the bundle is Up.

```
Router# show bundle bundle-ether 3
Tue Feb  4 18:24:25.313 UTC
```

```
Bundle-Ether1
```

```
Status: Up
Local links <active/standby/configured>: 3 / 0 / 3
Local bandwidth <effective/available>: 30000000 (30000000) kbps
MAC address (source): 1234.1234.1234 (Configured)
Inter-chassis link: No
Minimum active links / bandwidth: 1 / 1 kbps
Maximum active links: 32
Wait while timer: 2000 ms
Load balancing: Default
LACP: Not operational
  Flap suppression timer: Off
  Cisco extensions: Disabled
  Non-revertive: Disabled
mLACP: Not configured
IPv4 BFD: Not configured
```

Port	Device	State	Port ID	B/W, kbps
Tel1/0/0/0	Local	Active	0x8000, 0x0000	10000000
Link is Active				
Tel1/0/0/1	Local	Active	0x8000, 0x0000	10000000
Link is Active				
Tel1/0/0/2	Local	Active	0x8000, 0x0000	10000000
Link is Active				

Associated Commands

- [bundle maximum-active links](#)
- [interface Bundle-Ether](#)

- [show bundle Bundle-Ether](#)

Configure VLAN Bundle

The procedure for creating VLAN bundle is the same as the procedure for creating VLAN sub-interfaces on a physical ethernet interface.

Configuration Example

To configure VLAN bundles, complete the following configurations:

- Create a bundle instance.
- Create a VLAN interface (bundle sub-interface).
- Map the physical interface(s) to the bundle.

For a VLAN bundle to be active, you must perform the same configuration on both end points of the VLAN bundle.

Configuration

```
/* Enter global configuration mode and create VLAN bundle */
Router# configure
Router(config)# interface Bundle-Ether 2
Router(config-if)# ipv4 address 50.0.0.1/24
Router(config-if)# bundle maximum-active links 32 hot-standby
Router(config-if)# bundle minimum-active bandwidth 30000000
Router(config-if)# bundle minimum-active links 1
Router(config-if)# commit

/* Create VLAN sub-interface and add to the bundle */
Router(config)# interface Bundle-Ether 2.201
Router(config-subif)# ipv4 address 12.22.1.1 255.255.255.0
Router(config-subif)# encapsulation dot1q 201
Router(config-subif)# commit

/* Map the physical interface to the bundle */
Router(config)# interface TenGigE 0/0/0/14
Router(config-if)# bundle id 2 mode on
Router(config-if)# no shutdown
Router(config-if)# commit

/* Repeat the above steps for all the member interfaces:
0/0/0/15, 0/0/0/16 and 0/0/0/17 in this example */
```

Running Configuration

```
configure
interface Bundle-Ether2
  ipv4 address 50.0.0.1 255.255.255.0
  mac-address 1212.1212.1212
  bundle maximum-active links 32 hot-standby
  bundle minimum-active links 1
  bundle minimum-active bandwidth 30000000
!
```

```
interface Bundle-Ether2.201
  ipv4 address 12.22.1.1 255.255.255.0
  encapsulation dot1q 201
  !
interface TenGigE0/0/0/14
  bundle id 2 mode on
  !
interface TenGigE0/0/0/15
  bundle id 2 mode on
  !
interface TenGigE0/0/0/16
  bundle id 2 mode on
  !
interface TenGigE0/0/0/17
  bundle id 2 mode on
  !
```

Verification

Verify that the VLAN status is UP.

```
Router# show interfaces bundle-ether 2.201
```

```
Wed Feb  5 17:19:53.964 UTC
Bundle-Ether2.201 is up, line protocol is up
  Interface state transitions: 1
  Hardware is VLAN sub-interface(s), address is 28c7.ce01.dc7b
  Internet address is 12.22.1.1/24
  MTU 1518 bytes, BW 20000000 Kbit (Max: 20000000 Kbit)
    reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation 802.1Q Virtual LAN, VLAN Id 201,  loopback not set,
  Last link flapped 07:45:25
  ARP type ARPA, ARP timeout 04:00:00
  Last input 00:00:00, 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
    2938 packets input, 311262 bytes, 0 total input drops
  - - -
  - - -
```

Associated Commands

- [bundle maximum-active links](#)
- [interface Bundle-Ether](#)
- [show bundle Bundle-Ether](#)

References for Configuring Link Bundles

This section provides references to configuring link bundles. For an overview of link bundles and configurations, see [Configure Link Bundles for Layer 2 VPNs, on page 33](#).

Characteristics of Link Bundles

- Any type of Ethernet interfaces can be bundled, with or without the use of LACP (Link Aggregation Control Protocol).
- 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.
- A bundle can be administratively enabled or disabled.
- Each individual link within a bundle can be administratively enabled or disabled.
- Ethernet link bundles are created in the same way as Etherchannel channels, where the user enters the same configuration on both end systems.
- The MAC address that is set on the bundle becomes the MAC address of the links within that bundle.
- When LACP configured, each link within a bundle can be configured to allow different keepalive periods on different members.
- Load balancing 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, work independently on each link within a bundle.
- Upper layer protocols, such as routing updates and hello messages, are sent over any member link of an interface bundle.
- Bundled interfaces are point to point.
- A link must be in the UP state before it can be in distributing state in a bundle.
- 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.

Methods of Forming Bundles of Ethernet Interfaces

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.

For each link configured as bundle member, 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 routers 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 It is recommended that you avoid modifying the MAC address, because changes in the MAC address can affect packet forwarding.

- EtherChannel—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.

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 a router, the peer can be either another router or a switch. LACP monitors the operational state of link bundles to ensure these:

- 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.



CHAPTER 5

Configure Multipoint Layer 2 Services

This module provides the conceptual and configuration information for Multipoint Layer 2 Bridging Services, also called Virtual Private LAN Services (VPLS).



Note VPLS supports Layer 2 VPN technology and provides transparent multipoint Layer 2 connectivity for customers. This approach enables service providers to host a multitude of new services such as broadcast TV and Layer 2 VPNs.

- [Prerequisites for Implementing Multipoint Layer 2 Services, on page 41](#)
- [Information About Implementing Multipoint Layer 2 Services, on page 42](#)
- [Configuration Examples for Multipoint Layer 2 Services, on page 48](#)

Prerequisites for Implementing Multipoint Layer 2 Services

Before configuring Multipoint Layer 2 Services, ensure that these tasks and conditions are met:

- 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.
- Configure IP routing in the core so that the provider edge (PE) routers can reach each other through IP.
- Configure a loopback interface to originate and terminate Layer 2 traffic. Make sure that the PE routers can access the other router's loopback interface.



Note The loopback interface is not needed in all cases. For example, tunnel selection does not need a loopback interface when Multipoint Layer 2 Services are directly mapped to a TE tunnel.

Information About Implementing Multipoint Layer 2 Services

To implement Multipoint Layer 2 Services, you must understand these concepts:

Multipoint Layer 2 Services Overview

Multipoint Layer 2 Services enable geographically separated local-area network (LAN) segments to be interconnected as a single bridged domain over an MPLS network. The full functions of the traditional LAN such as MAC address learning, aging, and switching are emulated across all the remotely connected LAN segments that are part of a single bridged domain. A service provider can offer VPLS service to multiple customers over the MPLS network by defining different bridged domains for different customers. Packets from one bridged domain are never carried over or delivered to another bridged domain, thus ensuring the privacy of the LAN service.

Some of the components present in a Multipoint Layer 2 Services network are described in these sections.



Note Multipoint Layer 2 services are also called as Virtual Private LAN Services.

Bridge Domain

The native bridge domain refers to a Layer 2 broadcast domain consisting of a set of physical or virtual ports (including VFI). Data frames are switched within a bridge domain based on the destination MAC address. Multicast, broadcast, and unknown destination unicast frames are flooded within the bridge domain. In addition, the source MAC address learning is performed on all incoming frames on a bridge domain. A learned address is aged out. Incoming frames are mapped to a bridge domain, based on either the ingress port or a combination of both an ingress port and a MAC header field.

Pseudowires

A pseudowire is a point-to-point connection between pairs of PE routers. Its primary function is to emulate services like Ethernet over an underlying core MPLS network through encapsulation into a common MPLS format. By encapsulating services into a common MPLS format, a pseudowire allows carriers to converge their services to an MPLS network.

Access Pseudowire is not supported over VPLS Bridge Domain

Access PW is not supported over VPLS bridge domain. Only core PW which is configured under VFI is supported.

Configuration Example

```
l2vpn
bridge group bg1
  bridge-domain l2vpn
  interface TenGigE0/0/0/13.100
  !
vfi 1
  neighbor 192.0.2.1 pw-id 12345
  pw-class mpls_csr
  !
```

!
!

Virtual Forwarding Instance

VPLS is based on the characteristic of virtual forwarding instance (VFI). A VFI is a virtual bridge port that is capable of performing native bridging functions, such as forwarding, based on the destination MAC address, source MAC address learning and aging, and so forth.

A VFI is created on the PE router for each VPLS instance. The PE routers make packet-forwarding decisions by looking up the VFI of a particular VPLS instance. The VFI acts like a virtual bridge for a given VPLS instance. More than one attachment circuit belonging to a given VPLS are connected to the VFI. The PE router establishes emulated VCs to all the other PE routers in that VPLS instance and attaches these emulated VCs to the VFI. Packet forwarding decisions are based on the data structures maintained in the VFI.

VPLS for an MPLS-based Provider Core

VPLS is a multipoint Layer 2 VPN technology that connects two or more customer devices using bridging techniques. A bridge domain, which is the building block for multipoint bridging, is present on each of the PE routers. The access connections to the bridge domain on a PE router are called attachment circuits. The attachment circuits can be a set of physical ports, virtual ports, or both that are connected to the bridge at each PE device in the network.

After provisioning attachment circuits, neighbor relationships across the MPLS network for this specific instance are established through a set of manual commands identifying the end PEs. When the neighbor association is complete, a full mesh of pseudowires is established among the network-facing provider edge devices, which is a gateway between the MPLS core and the customer domain.

The MPLS/IP provider core simulates a virtual bridge that connects the multiple attachment circuits on each of the PE devices together to form a single broadcast domain. This also requires all of the PE routers that are participating in a VPLS instance to form emulated virtual circuits (VCs) among them.

Now, the service provider network starts switching the packets within the bridged domain specific to the customer by looking at destination MAC addresses. All traffic with unknown, broadcast, and multicast destination MAC addresses is flooded to all the connected customer edge devices, which connect to the service provider network. The network-facing provider edge devices learn the source MAC addresses as the packets are flooded. The traffic is unicasted to the customer edge device for all the learned MAC addresses.

VPLS for Layer 2 Switching

VPLS technology includes the capability of configuring the router to perform Layer 2 bridging. In this mode, the router can be configured to operate like other Cisco switches.

The storm control that is applied to multiple subinterfaces of the same physical port pertains to that physical port only. All subinterfaces with storm control configured are policed as aggregate under a single policer rate shared by all EFPs. None of the subinterfaces are configured with a dedicated policer rate. When a storm occurs on several subinterfaces simultaneously, and because subinterfaces share the policer, you can slightly increase the policer rate to accommodate additional policing.

These features are supported:

- Bridging IOS XR Trunk Interfaces
- Bridging on EFPs

Interoperability Between Cisco IOS XR and Cisco IOS on VPLS LDP Signaling

The Cisco IOS Software encodes the NLRI length in the first byte in bits format in the BGP Update message. However, the Cisco IOS XR Software interprets the NLRI length in 2 bytes. Therefore, when the BGP neighbor with VPLS-VPWS address family is configured between the IOS and the IOS XR, NLRI mismatch can happen, leading to flapping between neighbors. To avoid this conflict, IOS supports **prefix-length-size 2** command that needs to be enabled for IOS to work with IOS XR. When the **prefix-length-size 2** command is configured in IOS, the NLRI length is encoded in bytes. This configuration is mandatory for IOS to work with IOS XR.

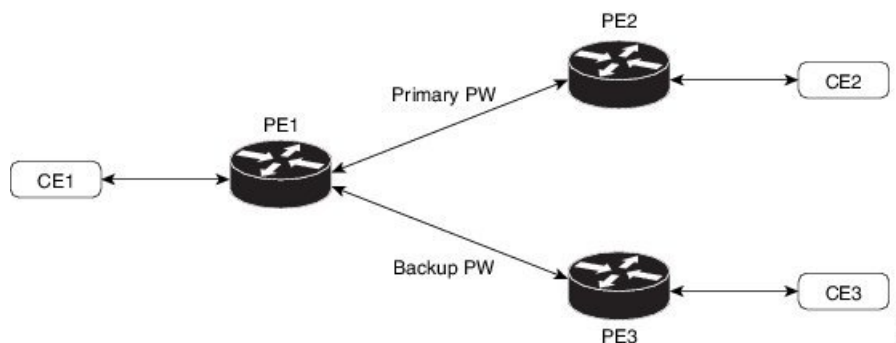
This is a sample IOS configuration with the **prefix-length-size 2** command:

```
router bgp 1
 address-family l2vpn vpls
  neighbor 5.5.5.2 activate
  neighbor 5.5.5.2 prefix-length-size 2 -----> NLRI length = 2 bytes
 exit-address-family
```

Pseudowire Redundancy

Pseudowire redundancy allows you to configure a backup pseudowire in case the primary pseudowire fails. When the primary pseudowire fails, the PE router can switch to the backup pseudowire. You can elect to have the primary pseudowire resume operation after it becomes functional. The primary pseudowire fails when the PE router fail or due to any network related outage.

Figure 3: Pseudowire Redundancy



Forcing a Manual Switchover to the Backup Pseudowire

To force the router to switch over to the backup or switch back to the primary pseudowire, use the **l2vpn switchover** command in EXEC mode.

A manual switchover is made only if the peer specified in the command is actually available and the cross-connect moves to the fully active state when the command is entered.

Configuration

This section describes how you can configure pseudowire redundancy.

```
/* Configure PE1 */
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group XCON1
Router(config-l2vpn-xc)# p2p xc1
```

```

Router(config-l2vpn-xc-p2p)# interface GigabitEthernet 0/1/0/0.1
Router(config-l2vpn-xc-p2p)# neighbor ipv4 2.2.2.2 pw-id 1
Router(config-l2vpn-xc-p2p-pw)# backup neighbor 3.3.3.3 pw-id 1
Router(config-subif)# commit

/* Configure PE2 */
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group XCON1
Router(config-l2vpn-xc)# p2p xcl
Router(config-l2vpn-xc-p2p)# interface GigabitEthernet 0/1/0/0.1
Router(config-l2vpn-xc-p2p)# neighbor ipv4 1.1.1.1 pw-id 1
Router(config-subif)# commit

/* Configure PE3 */
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group XCON1
Router(config-l2vpn-xc)# p2p xcl
Router(config-l2vpn-xc-p2p)# interface GigabitEthernet 0/1/0/0.1
Router(config-l2vpn-xc-p2p)# neighbor ipv4 1.1.1.1 pw-id 1
Router(config-subif)# commit

```

Running Configuration

```

/* On PE1 */
!
l2vpn
xconnect group XCON1
p2p XCON1_P2P2
interface GigabitEthernet 0/1/0/0.1
neighbor ipv4 2.2.2.2 pw-id 1
backup neighbor 3.3.3.3 pw-id 1
!

/* On PE2 */
!
l2vpn
xconnect group XCON1
p2p XCON1_P2P2
interface GigabitEthernet 0/1/0/0.1
neighbor ipv4 1.1.1.1 pw-id 1
!

/* On PE3 */
!
l2vpn
xconnect group XCON1
p2p XCON1_P2P2
interface GigabitEthernet 0/1/0/0.1
neighbor ipv4 1.1.1.1 pw-id 1
!

```

Verification

Verify that the configured pseudowire redundancy is up.

```

/* On PE1 */

Router#show l2vpn xconnect group XCON_1

```

Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect Group	Name	ST	Segment 1 Description	ST	Segment 2 Description	ST
XCON_1	XCON1_P2P2	UP	Gi0/1/0/0.1	UP	2.2.2.2 1000	UP
					Backup 3.3.3.3 1000	SB

/* On PE2 */

Router#show l2vpn xconnect group XCON_1

Tue Jan 17 15:36:12.327 UTC

Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect Group	Name	ST	Segment 1 Description	ST	Segment 2 Description	ST
XCON_1	XCON1_P2P2	UP	BE100.1	UP	1.1.1.1 1000	UP

/* On PE3 */

Router#show l2vpn xconnect group XCON_1

Tue Jan 17 15:38:04.785 UTC

Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect Group	Name	ST	Segment 1 Description	ST	Segment 2 Description	ST
XCON_1	XCON1_P2P2	DN	BE100.1	UP	1.1.1.1 1000	SB

Router#show l2vpn xconnect summary

Number of groups: 3950

Number of xconnects: 3950

Up: 3950 Down: 0 Unresolved: 0 Partially-programmed: 0

AC-PW: 3950 AC-AC: 0 PW-PW: 0 Monitor-Session-PW: 0

Number of Admin Down segments: 0

Number of MP2MP xconnects: 0

Up 0 Down 0

Advertised: 0 Non-Advertised: 0

Number of CE Connections: 0

Advertised: 0 Non-Advertised: 0

Backup PW:

Configured : 3950

UP : 0

Down : 0

Admin Down : 0

Unresolved : 0

Standby : 3950

Standby Ready: 0

Backup Interface:

Configured : 0

UP : 0

Down : 0

Admin Down : 0

Unresolved : 0

Standby : 0

MAC Address-related Parameters

The MAC address table contains a list of the known MAC addresses and their forwarding information. In the current VPLS design, the MAC address table and its management are maintained on the route processor (RP) card.

These topics provide information about the MAC address-related parameters:

MAC Address Flooding

Ethernet services require that frames that are sent to broadcast addresses and to unknown destination addresses be flooded to all ports. To obtain flooding within VPLS broadcast models, all unknown unicast, broadcast, and multicast frames are flooded over the corresponding pseudowires and to all attachment circuits. Therefore, a PE must replicate packets across both attachment circuits and pseudowires.

MAC Address-based Forwarding

To forward a frame, a PE must associate a destination MAC address with a pseudowire or attachment circuit. This type of association is provided through a static configuration on each PE or through dynamic learning, which is flooded to all bridge ports.

MAC Address Source-based Learning

When a frame arrives on a bridge port (for example, pseudowire or attachment circuit) and the source MAC address is unknown to the receiving PE router, the source MAC address is associated with the pseudowire or attachment circuit. Outbound frames to the MAC address are forwarded to the appropriate pseudowire or attachment circuit.

MAC address source-based learning uses the MAC address information that is learned in the hardware forwarding path. The updated MAC tables are propagated and programs the hardware for the router.



Note Static MAC move is not supported from one port, interface, or AC to another port, interface, or AC. For example, if a static MAC is configured on AC1 (port 1) and then, if you send a packet with the same MAC as source MAC on AC2 (port 2), then you can't attach this MAC to AC2 as a dynamic MAC. Therefore, do not send any packet with a MAC as any of the static MAC addresses configured.

The number of learned MAC addresses is limited through configurable per-port and per-bridge domain MAC address limits.

MAC Address Aging

A MAC address in the MAC table is considered valid only for the duration of the MAC address aging time. When the time expires, the relevant MAC entries are repopulated. When the MAC aging time is configured only under a bridge domain, all the pseudowires and attachment circuits in the bridge domain use that configured MAC aging time.

A bridge forwards, floods, or drops packets based on the bridge table. The bridge table maintains both static entries and dynamic entries. Static entries are entered by the network manager or by the bridge itself. Dynamic entries are entered by the bridge learning process. A dynamic entry is automatically removed after a specified length of time, known as *aging time*, from the time the entry was created or last updated.

If hosts on a bridged network are likely to move, decrease the aging-time to enable the bridge to adapt to the change quickly. If hosts do not transmit continuously, increase the aging time to record the dynamic entries for a longer time, thus reducing the possibility of flooding when the hosts transmit again.

MAC Address Limit

The MAC address limit is used to limit the number of learned MAC addresses.

When a limit is exceeded, the system is configured to perform these notifications:

- Syslog (default)
- Simple Network Management Protocol (SNMP) trap
- Syslog and SNMP trap
- None (no notification)



Note Though you can modify the MAC address limit under the bridge domain, due to hardware limitation, the modification does not take effect.

MAC Address Withdrawal

For faster VPLS convergence, you can remove or unlearn the MAC addresses that are learned dynamically. The Label Distribution Protocol (LDP) Address Withdrawal message is sent with the list of MAC addresses, which need to be withdrawn to all other PEs that are participating in the corresponding VPLS service.

For the Cisco IOS XR VPLS implementation, a portion of the dynamically learned MAC addresses are cleared by using the MAC addresses aging mechanism by default. The MAC address withdrawal feature is added through the LDP Address Withdrawal message. To enable the MAC address withdrawal feature, use the **withdrawal** command in l2vpn bridge group bridge domain MAC configuration mode. To verify that the MAC address withdrawal is enabled, use the **show l2vpn bridge-domain** command with the **detail** keyword.



Note By default, the LDP MAC Withdrawal feature is enabled on Cisco IOS XR.

The LDP MAC Withdrawal feature is generated due to these events:

- Attachment circuit goes down. You can remove or add the attachment circuit through the CLI.
- MAC withdrawal messages are received over a VFI pseudowire. RFC 4762 specifies that both wildcards (by means of an empty Type, Length and Value [TLV]) and a specific MAC address withdrawal. Cisco IOS XR software supports only a wildcard MAC address withdrawal.

Configuration Examples for Multipoint Layer 2 Services

This section includes these configuration examples:

Multipoint Layer 2 Services Configuration for Provider Edge-to-Provider Edge: Example

These configuration examples show how to create a Layer 2 VFI with a full-mesh of participating Multipoint Layer 2 Services provider edge (PE) nodes.

This configuration example shows how to configure PE 1:

```
configure
l2vpn
bridge group 1
  bridge-domain PE1-VPLS-A
  interface TenGigE0/0/0/0
  vfi 1
    neighbor 10.2.2.2 pw-id 1
    neighbor 10.3.3.3 pw-id 1
  !
!
interface loopback 0
  ipv4 address 10.1.1.1 255.255.255.255
```

This configuration example shows how to configure PE 2:

```
configure
l2vpn
bridge group 1
  bridge-domain PE2-VPLS-A
  interface TenGigE0/0/0/1

  vfi 1
    neighbor 10.1.1.1 pw-id 1
    neighbor 10.3.3.3 pw-id 1
  !
!
interface loopback 0
  ipv4 address 10.2.2.2 255.255.255.255
```

This configuration example shows how to configure PE 3:

```
configure
l2vpn
bridge group 1
  bridge-domain PE3-VPLS-A
  interface TenGigE0/0/0/2
  vfi 1
    neighbor 10.1.1.1 pw-id 1
    neighbor 10.2.2.2 pw-id 1
  !
!
interface loopback 0
  ipv4 address 10.3.3.3 255.255.255.255
```

Multipoint Layer 2 Services Configuration for Provider Edge-to-Customer Edge: Example

This configuration shows how to configure Multipoint Layer 2 Services for a PE-to-CE nodes:

```
configure
interface TenGigE0/0/0/0
  l2transport---AC interface
```

Displaying MAC Address Withdrawal Fields: Example

```
no ipv4 address
no ipv4 directed-broadcast
negotiation auto
no cdp enable
```

Displaying MAC Address Withdrawal Fields: Example

This sample output shows the MAC address withdrawal fields:

```
RP/0/RSP0/CPU0:router# show l2vpn bridge-domain detail
```

```
Legend: pp = Partially Programmed.
Bridge group: 222, bridge-domain: 222, id: 0, state: up, ShgId: 0, MSTi: 0
  Coupled state: disabled
  MAC learning: enabled
  MAC withdraw: enabled
    MAC withdraw sent on: bridge port up
    MAC withdraw relaying (access to access): disabled
  Flooding:
    Broadcast & Multicast: enabled
    Unknown unicast: enabled
  MAC aging time: 300 s, Type: inactivity
  MAC limit: 4000, Action: none, Notification: syslog
  MAC limit reached: no
  MAC port down flush: enabled
  MAC Secure: disabled, Logging: disabled
  Split Horizon Group: none
  Dynamic ARP Inspection: disabled, Logging: disabled
  IP Source Guard: disabled, Logging: disabled
  DHCPv4 snooping: disabled
  IGMP Snooping: enabled
  IGMP Snooping profile: none
  MLD Snooping profile: none
  Storm Control: disabled
  Bridge MTU: 1500
  MIB cvplsConfigIndex: 1
  Filter MAC addresses:
  P2MP PW: disabled
  Create time: 01/03/2017 11:01:11 (00:21:33 ago)
  No status change since creation
  ACs: 1 (1 up), VFIs: 1, PWs: 1 (1 up), PBBs: 0 (0 up)
  List of ACs:
    AC: TenGigE0/2/0/1.7, state is up
      Type VLAN; Num Ranges: 1
      Outer Tag: 21
      VLAN ranges: [22, 22]
      MTU 1508; XC ID 0x208000b; interworking none
      MAC learning: enabled
      Flooding:
        Broadcast & Multicast: enabled
        Unknown unicast: enabled
      MAC aging time: 300 s, Type: inactivity
      MAC limit: 4000, Action: none, Notification: syslog
      MAC limit reached: no
      MAC port down flush: enabled
      MAC Secure: disabled, Logging: disabled
      Split Horizon Group: none
      Dynamic ARP Inspection: disabled, Logging: disabled
      IP Source Guard: disabled, Logging: disabled
      DHCPv4 snooping: disabled
      IGMP Snooping: enabled
```

```

IGMP Snooping profile: none
MLD Snooping profile: none
Storm Control: bridge-domain policer
Static MAC addresses:
Statistics:
  packets: received 714472608 (multicast 0, broadcast 0, unknown unicast 0, unicast
0), sent 97708776
  bytes: received 88594603392 (multicast 0, broadcast 0, unknown unicast 0, unicast
0), sent 12115888224
  MAC move: 0
Storm control drop counters:
  packets: broadcast 0, multicast 0, unknown unicast 0
  bytes: broadcast 0, multicast 0, unknown unicast 0
Dynamic ARP inspection drop counters:
  packets: 0, bytes: 0
IP source guard drop counters:
  packets: 0, bytes: 0
List of VFIIs:
VFI 222 (up)
PW: neighbor 1.1.1.1, PW ID 222, state is up ( established )
PW class not set, XC ID 0xc000000a
Encapsulation MPLS, protocol LDP
Source address 21.21.21.21
PW type Ethernet, control word disabled, interworking none
Sequencing not set

PW Status TLV in use
MPLS          Local                               Remote
-----
Label         24017                                           24010
Group ID      0x0                                             0x0
Interface     222                                             222
MTU           1500                                           1500
Control word  disabled                                       disabled
PW type       Ethernet                                       Ethernet
VCCV CV type 0x2
              (LSP ping verification)                   (LSP ping verification)
VCCV CC type 0x6
              (router alert label)                       (router alert label)
              (TTL expiry)                             (TTL expiry)
-----

Incoming Status (PW Status TLV):
  Status code: 0x0 (Up) in Notification message
MIB cpwVcIndex: 3221225482
Create time: 01/03/2017 11:01:11 (00:21:33 ago)
Last time status changed: 01/03/2017 11:21:01 (00:01:43 ago)
Last time PW went down: 01/03/2017 11:15:21 (00:07:23 ago)
MAC withdraw messages: sent 0, received 0
Forward-class: 0
Static MAC addresses:
Statistics:
  packets: received 95320440 (unicast 0), sent 425092569
  bytes: received 11819734560 (unicast 0), sent 52711478556
  MAC move: 0
Storm control drop counters:
  packets: broadcast 0, multicast 0, unknown unicast 0
  bytes: broadcast 0, multicast 0, unknown unicast 0
DHCPv4 snooping: disabled
IGMP Snooping profile: none
MLD Snooping profile: none
VFI Statistics:
  drops: illegal VLAN 0, illegal length 0

```

Bridging on IOS XR Trunk Interfaces: Example

This example shows how to configure a Cisco NCS 5000 Series Routers as a simple L2 switch.

Important notes:

Create a bridge domain that has four attachment circuits (AC). Each AC is an IOS XR trunk interface (i.e. not a subinterface/EFP).

- This example assumes that the running config is empty, and that all the components are created.
- This example provides all the necessary steps to configure the Cisco NCS 5000 Series Routers to perform switching between the interfaces. However, the commands to prepare the interfaces such as no shut, negotiation auto, etc., have been excluded.
- The bridge domain is in a no shut state, immediately after being created.
- Only trunk (i.e. main) interfaces are used in this example.
- The trunk interfaces are capable of handling tagged (i.e. IEEE 802.1Q) or untagged (i.e. no VLAN header) frames.
- The bridge domain learns, floods, and forwards based on MAC address. This functionality works for frames regardless of tag configuration.
- The bridge domain entity spans the entire system. It is not necessary to place all the bridge domain ACs on a single LC. This applies to any bridge domain configuration.
- The show bundle and the show l2vpn bridge-domain commands are used to verify that the router was configured as expected, and that the commands show the status of the new configurations.
- The ACs in this example use interfaces that are in the admin down state.

Configuration Example

```
RP/0/RSP0/CPU0:router#config
RP/0/RSP0/CPU0:router(config)#interface Bundle-ether10
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#interface GigabitEthernet0/2/0/5
RP/0/RSP0/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP0/CPU0:router(config-if)#interface GigabitEthernet0/2/0/6
RP/0/RSP0/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP0/CPU0:router(config-if)#interface GigabitEthernet0/2/0/0
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#interface GigabitEthernet0/2/0/1
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#interface TenGigE0/1/0/2
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)#bridge group examples
RP/0/RSP0/CPU0:router(config-l2vpn-bg)#bridge-domain test-switch
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface Bundle-ether10
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface GigabitEthernet0/2/0/0
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface GigabitEthernet0/2/0/1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface TenGigE0/1/0/2
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#commit
RP/0/RSP0/CPU0:Jul 26 10:48:21.320 EDT: config[65751]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'lab'. Use 'show configuration commit changes 1000000973'
to view the changes.
```

```

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#end
RP/0/RSP0/CPU0:Jul 26 10:48:21.342 EDT: config[65751]: %MGBL-SYS-5-CONFIG_I : Configured
from console by lab
RP/0/RSP0/CPU0:router#show bundle Bundle-ether10

Bundle-Ether10
  Status:                               Down
  Local links <active/standby/configured>: 0 / 0 / 2
  Local bandwidth <effective/available>:    0 (0) kbps
  MAC address (source):                    0024.f71e.22eb (Chassis pool)
  Minimum active links / bandwidth:        1 / 1 kbps
  Maximum active links:                     64
  Wait while timer:                         2000 ms
  LACP:                                     Operational
    Flap suppression timer:                 Off
  mLACP:                                    Not configured
  IPv4 BFD:                                 Not configured

Port          Device          State          Port ID          B/W, kbps
-----
Gi0/2/0/5     Local           Configured     0x8000, 0x0001  1000000
  Link is down
Gi0/2/0/6     Local           Configured     0x8000, 0x0002  1000000
  Link is down

RP/0/RSP0/CPU0:router#
RP/0/RSP0/CPU0:router#show l2vpn bridge-domain group examples
Bridge group: examples, bridge-domain: test-switch, id: 2000, state: up, ShgId: 0, MSTi: 0
Aging: 300 s, MAC limit: 4000, Action: none, Notification: syslog
Filter MAC addresses: 0
ACs: 4 (1 up), VFIs: 0, PWs: 0 (0 up), PBBs: 0 (0 up)
List of ACs:
  BE10, state: down, Static MAC addresses: 0
  Gi0/2/0/0, state: up, Static MAC addresses: 0
  Gi0/2/0/1, state: down, Static MAC addresses: 0
  Te0/5/0/1, state: down, Static MAC addresses: 0
List of VFIs:
RP/0/RSP0/CPU0:router#

```

This table lists the configuration steps (actions) and the corresponding purpose for this example:

SUMMARY STEPS

1. **configure**
2. **interface Bundle-ether10**
3. **l2transport**
4. **interface GigabitEthernet0/2/0/5**
5. **bundle id 10 mode active**
6. **interface GigabitEthernet0/2/0/6**
7. **bundle id 10 mode active**
8. **interface GigabitEthernet0/2/0/0**
9. **l2transport**
10. **interface GigabitEthernet0/2/0/1**
11. **l2transport**
12. **interface TenGigE0/1/0/2**
13. **l2transport**
14. **l2vpn**

15. **bridge group examples**
16. **bridge-domain test-switch**
17. **interface Bundle-ether10**
18. **exit**
19. **interface GigabitEthernet0/2/0/0**
20. **exit**
21. **interface GigabitEthernet0/2/0/1**
22. **exit**
23. **interface TenGigE0/1/0/2**
24. Use the **commit** or **end** command.

DETAILED STEPS

- | | |
|----------------|---|
| Step 1 | configure
Enters global configuration mode. |
| Step 2 | interface Bundle-ether10
Creates a new bundle trunk interface. |
| Step 3 | l2transport
Changes Bundle-ether10 from an L3 interface to an L2 interface. |
| Step 4 | interface GigabitEthernet0/2/0/5
Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/5. |
| Step 5 | bundle id 10 mode active
Establishes GigabitEthernet0/2/0/5 as a member of Bundle-ether10. The mode active keywords specify LACP protocol. |
| Step 6 | interface GigabitEthernet0/2/0/6
Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/6. |
| Step 7 | bundle id 10 mode active
Establishes GigabitEthernet0/2/0/6 as a member of Bundle-ether10. The mode active keywords specify LACP protocol. |
| Step 8 | interface GigabitEthernet0/2/0/0
Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/0. |
| Step 9 | l2transport
Change GigabitEthernet0/2/0/0 from an L3 interface to an L2 interface. |
| Step 10 | interface GigabitEthernet0/2/0/1
Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/1. |
| Step 11 | l2transport
Change GigabitEthernet0/2/0/1 from an L3 interface to an L2 interface. |

- Step 12** **interface TenGigE0/1/0/2**
Enters interface configuration mode. Changes configuration mode to act on TenGigE0/1/0/2.
- Step 13** **l2transport**
Changes TenGigE0/1/0/2 from an L3 interface to an L2 interface.
- Step 14** **l2vpn**
Enters L2VPN configuration mode.
- Step 15** **bridge group examples**
Creates the bridge group **examples**.
- Step 16** **bridge-domain test-switch**
Creates the bridge domain **test-switch**, that is a member of bridge group **examples**.
- Step 17** **interface Bundle-ether10**
Establishes Bundle-ether10 as an AC of bridge domain **test-switch**.
- Step 18** **exit**
Exits bridge domain AC configuration submode, allowing next AC to be configured.
- Step 19** **interface GigabitEthernet0/2/0/0**
Establishes GigabitEthernet0/2/0/0 as an AC of bridge domain **test-switch**.
- Step 20** **exit**
Exits bridge domain AC configuration submode, allowing next AC to be configured.
- Step 21** **interface GigabitEthernet0/2/0/1**
Establishes GigabitEthernet0/2/0/1 as an AC of bridge domain **test-switch**.
- Step 22** **exit**
Exits bridge domain AC configuration submode, allowing next AC to be configured.
- Step 23** **interface TenGigE0/1/0/2**
Establishes interface TenGigE0/1/0/2 as an AC of bridge domain **test-switch**.
- Step 24** Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
- **Yes** - Saves configuration changes and exits the configuration session.
 - **No** - Exits the configuration session without committing the configuration changes.
 - **Cancel** - Remains in the configuration mode, without committing the configuration changes.
-

Bridging on Ethernet Flow Points: Example

This example shows how to configure a Cisco NCS 5000 Series Router to perform Layer 2 switching on traffic that passes through Ethernet Flow Points (EFPs). EFP traffic typically has one or more VLAN headers. Although both IOS XR trunks and IOS XR EFPs can be combined as attachment circuits in bridge domains, this example uses EFPs exclusively.

Important notes:

- An EFP is a Layer 2 subinterface. It is always created under a trunk interface. The trunk interface must exist before the EFP is created.
- In an empty configuration, the bundle interface trunk does not exist, but the physical trunk interfaces are automatically configured. Therefore, only the bundle trunk is created.
- In this example the subinterface number and the VLAN IDs are identical, but this is out of convenience, and is not a necessity. They do not need to be the same values.
- The bridge domain test-efp has three attachment circuits (ACs). All the ACs are EFPs.
- Only frames with a VLAN ID of 999 enter the EFPs. This ensures that all the traffic in this bridge domain has the same VLAN encapsulation.
- The ACs in this example use interfaces that are in the admin down state (**unresolved** state). Bridge domains that use nonexistent interfaces as ACs are legal, and the commit for such configurations does not fail. In this case, the status of the bridge domain shows **unresolved** until you configure the missing interface.

Configuration Example

```
RP/0/RSP1/CPU0:router#configure
RP/0/RSP1/CPU0:router(config)#interface Bundle-ether10
RP/0/RSP1/CPU0:router(config-if)#interface Bundle-ether10.999 l2transport
RP/0/RSP1/CPU0:router(config-subif)#encapsulation dot1q 999
RP/0/RSP1/CPU0:router(config-subif)#interface GigabitEthernet0/6/0/5
RP/0/RSP1/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP1/CPU0:router(config-if)#interface GigabitEthernet0/6/0/6
RP/0/RSP1/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP1/CPU0:router(config-if)#interface GigabitEthernet0/6/0/7.999 l2transport
RP/0/RSP1/CPU0:router(config-subif)#encapsulation dot1q 999
RP/0/RSP1/CPU0:router(config-subif)#interface TenGigE0/1/0/2.999 l2transport
RP/0/RSP1/CPU0:router(config-subif)#encapsulation dot1q 999
RP/0/RSP1/CPU0:router(config-subif)#l2vpn
RP/0/RSP1/CPU0:router(config-l2vpn)#bridge group examples
RP/0/RSP1/CPU0:router(config-l2vpn-bg)#bridge-domain test-efp
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd)#interface Bundle-ether10.999
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd)#interface GigabitEthernet0/6/0/7.999
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd)#interface TenGigE0/1/0/2.999
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#commit
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#end
RP/0/RSP1/CPU0:router#
RP/0/RSP1/CPU0:router#show l2vpn bridge group examples
Fri Jul 23 21:56:34.473 UTC Bridge group: examples, bridge-domain: test-efp, id: 0, state:
up, ShgId: 0, MSTi: 0
Aging: 300 s, MAC limit: 4000, Action: none, Notification: syslog
Filter MAC addresses: 0
ACs: 3 (0 up), VFIs: 0, PWs: 0 (0 up), PBBs: 0 (0 up)
List of ACs:
```

```

BE10.999, state: down, Static MAC addresses: 0
Gi0/6/0/7.999, state: unresolved, Static MAC addresses: 0
Te0/1/0/2.999, state: down, Static MAC addresses: 0
List of VFIs:
RP/0/RSP1/CPU0:router#

```

This table lists the configuration steps (actions) and the corresponding purpose for this example:

SUMMARY STEPS

1. **configure**
2. **interface Bundle-ether10**
3. **interface Bundle-ether10.999 l2transport**
4. **encapsulation dot1q 999**
5. **interface GigabitEthernet0/6/0/5**
6. **bundle id 10 mode active**
7. **interface GigabitEthernet0/6/0/6**
8. **bundle id 10 mode active**
9. **interface GigabitEthernet0/6/0/7.999 l2transport**
10. **encapsulation dot1q 999**
11. **interface TenGigE0/1/0/2.999 l2transport**
12. **encapsulation dot1q 999**
13. **l2vpn**
14. **bridge group examples**
15. **bridge-domain test-efp**
16. **interface Bundle-ether10.999**
17. **exit**
18. **interface GigabitEthernet0/6/0/7.999**
19. **exit**
20. **interface TenGigE0/1/0/2.999**
21. Use the **commit** or **end** command.

DETAILED STEPS

-
- | | |
|---------------|---|
| Step 1 | configure
Enters global configuration mode. |
| Step 2 | interface Bundle-ether10
Creates a new bundle trunk interface. |
| Step 3 | interface Bundle-ether10.999 l2transport
Creates an EFP under the new bundle trunk. |
| Step 4 | encapsulation dot1q 999
Assigns VLAN ID of 999 to this EFP. |
| Step 5 | interface GigabitEthernet0/6/0/5 |

Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/6/0/5.

Step 6 **bundle id 10 mode active**

Establishes GigabitEthernet0/6/0/5 as a member of Bundle-ether10. The **mode active** keywords specify LACP protocol.

Step 7 **interface GigabitEthernet0/6/0/6**

Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/6/0/6.

Step 8 **bundle id 10 mode active**

Establishes GigabitEthernet0/6/0/6 as a member of Bundle-ether10. The **mode active** keywords specify LACP protocol.

Step 9 **interface GigabitEthernet0/6/0/7.999 l2transport**

Creates an EFP under GigabitEthernet0/6/0/7.

Step 10 **encapsulation dot1q 999**

Assigns VLAN ID of 999 to this EFP.

Step 11 **interface TenGigE0/1/0/2.999 l2transport**

Creates an EFP under TenGigE0/1/0/2.

Step 12 **encapsulation dot1q 999**

Assigns VLAN ID of 999 to this EFP.

Step 13 **l2vpn**

Enters L2VPN configuration mode.

Step 14 **bridge group examples**

Creates the bridge group named **examples**.

Step 15 **bridge-domain test-efp**

Creates the bridge domain named **test-efp**, that is a member of bridge group **examples**.

Step 16 **interface Bundle-ether10.999**

Establishes Bundle-ether10.999 as an AC of the bridge domain named **test-efp**.

Step 17 **exit**

Exits bridge domain AC configuration submode, allowing next AC to be configured.

Step 18 **interface GigabitEthernet0/6/0/7.999**

Establishes GigabitEthernet0/6/0/7.999 as an AC of the bridge domain named **test-efp**.

Step 19 **exit**

Exits bridge domain AC configuration submode, allowing next AC to be configured.

Step 20 **interface TenGigE0/1/0/2.999**

Establishes interface TenGigE0/1/0/2.999 as an AC of bridge domain named **test-efp**.

Step 21 Use the **commit** or **end** command.

commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
 - **No** - Exits the configuration session without committing the configuration changes.
 - **Cancel** - Remains in the configuration mode, without committing the configuration changes.
-



CHAPTER 6

Configure L2VPN Autodiscovery and Signaling

This chapter describes the L2VPN Autodiscovery and Signaling feature which enables the discovery of remote Provider Edge (PE) routers and the associated signaling in order to provision the pseudowires.

- [L2VPN Autodiscovery and Signaling, on page 61](#)
- [BGP-based VPLS Autodiscovery, on page 61](#)
- [BGP-based VPWS Autodiscovery, on page 66](#)

L2VPN Autodiscovery and Signaling

Autodiscovery refers to the process of finding the Provider Edge (PE) routers that participates in a given L2VPN instance. One of the protocols used for this is BGP.

Once the PE routers are discovered, pseudowires are signaled and established across each pair of PE routers. Signaling refers to the exchange of Virtual Circuit (VC) labels between the PE routers. The signaling protocol can be either LDP or BGP.

BGP-based VPLS Autodiscovery

VPLS is a multipoint Layer 2 bridging service for which BGP-based autodiscovery is well suited. BGP-based VPLS autodiscovery eliminates the need to manually provision the VPLS neighbors. VPLS autodiscovery enables each VPLS PE router to discover the other provider edge (PE) routers that are part of the same VPLS domain. VPLS Autodiscovery also tracks when PE routers are added to or removed from the VPLS domain. When the discovery process is complete, each PE router has the information required to setup VPLS pseudowires (PWs).

Even when BGP autodiscovery is enabled, pseudowires can be manually configured for VPLS PE routers that are not participating in the autodiscovery process.

BGP-based VPLS Autodiscovery with BGP Signaling

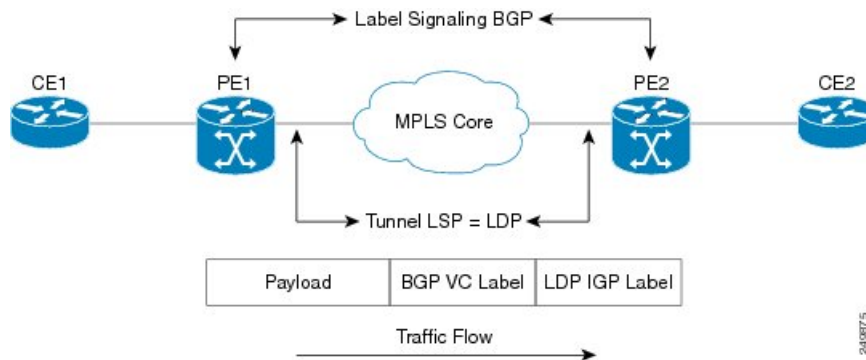
The BGP signaling and autodiscovery scheme have the following components:

- A means for a PE to learn which remote PEs are members of a given VPLS. This process is known as autodiscovery.

- A means for a PE to learn the pseudowire label expected by a given remote PE for a given VPLS. This process is known as signaling.

The BGP Network Layer Reachability Information (NLRI) takes care of the above two components simultaneously. The NLRI generated by a given PE contains the necessary information required by any other PE. These components enable the automatic setting up of a full mesh of pseudowires for each VPLS without having to manually configure those pseudowires on each PE.

Figure 4: Discovery and Signaling Attributes



Configuring BGP and LDP for BGP-based Autodiscovery

This is the basic BGP and LDP configuration that is required before proceeding to configure BGP-based autodiscovery.

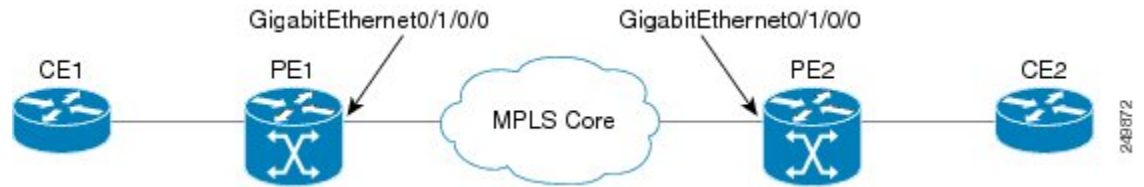
Configuration Example:

```
Router(config)# interface loopback-interface
Router(config-if)# ipv4 address ipv4-address subnet-mask
Router(config-if)# exit
Router(config)# mpls ldp
Router(config-ldp)# router-id ipv4-address
Router(config-ldp-if)# interface interface-name
Router(config-ldp-if)# exit
Router(config-ldp)# exit
Router(config)# router bgp as-number
Router(config-bgp)# address-family l2vpn vpls-vpws
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor loopback ipv4 address of neighbor
Router(config-bgp-nbr)# remote-as remote-as-number
Router(config-bgp-nbr)# update-source loopback-interface
Router(config-bgp-nbr)# address-family l2vpn vpls-vpws
```

Running Configuration

The following figure illustrates an example of LDP and BGP network topology that is required for enabling BGP based autodiscovery.

Figure 5: LDP and BGP Configuration Example

**Configuration at PE1:**

```
interface Loopback1
  ipv4 address 1.1.1.10 255.255.255.255
!
mpls ldp
  router-id 1.1.1.1
  interface GigabitEthernet0/1/0/0
!
router bgp 120
  address-family l2vpn vpls-vpws
!
  neighbor 2.2.2.20
  remote-as 120
  update-source Loopback1
  address-family l2vpn vpls-vpws
```

Configuration at PE2:

```
interface Loopback1
  ipv4 address 2.2.2.20 255.255.255.255
!
mpls ldp
  router-id 2.2.2.2
  interface GigabitEthernt0/1/0/0
!
router bgp 120
  address-family l2vpn vpls-vpws
!
  neighbor 1.1.1.10
  remote-as 120
  update-source Loopback1
  address-family l2vpn vpls-vpws
```

Configuring BGP-based VPLS Autodiscovery with BGP Signaling

BGP and LDP need to be configured as indicated in the section [Configuring BGP and LDP for BGP-based Autodiscovery, on page 62](#) before proceeding to the configurations in this section.

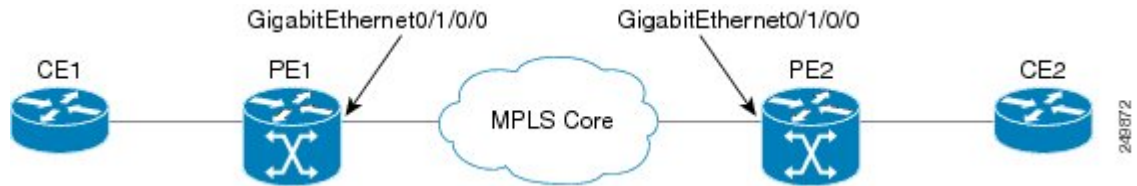
Configuration Example:

```
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bridge-group-name
Router(config-l2vpn-bg)# bridge-domain bridge-domain-name
Router(config-l2vpn-bg-bd)# vfi vfi-name
Router(config-l2vpn-bg-bd-vfi)# autodiscovery bgp
Router(config-l2vpn-bg-bd-vfi-ad)# vpn-id vpn-id
Router(config-l2vpn-bg-bd-vfi-ad)# rd auto
```

```
Router(config-l2vpn-bg-bd-vfi-ad)# route-target 1.1.1.1:100
Router(config-l2vpn-bg-bd-vfi-ad-sig)# signaling-protocol bgp
Router(config-l2vpn-bg-bd-vfi-ad-sig)# ve-id 1
```

Running Configuration:

Figure 6: BGP-based VPLS Autodiscovery with BGP signaling



Configuration at PE1:

```
l2vpn
  bridge group gr1
  bridge-domain bd1
  interface GigabitEthernet0/1/0/1.1
  vfi vfl
  ! AD independent VFI attributes
  vpn-id 100
  ! Auto-discovery attributes
  autodiscovery bgp
  rd auto
  route-target 2.2.2.2:100
  ! Signaling attributes
  signaling-protocol bgp
  ve-id 3
```

Configuration at PE2:

```
l2vpn
  bridge group gr1
  bridge-domain bd1
  interface GigabitEthernet0/1/0/2.1
  vfi vfl
  ! AD independent VFI attributes
  vpn-id 100
  ! Auto-discovery attributes
  autodiscovery bgp
  rd auto
  route-target 2.2.2.2:100
  ! Signaling attributes
  signaling-protocol bgp
  ve-id 5
```

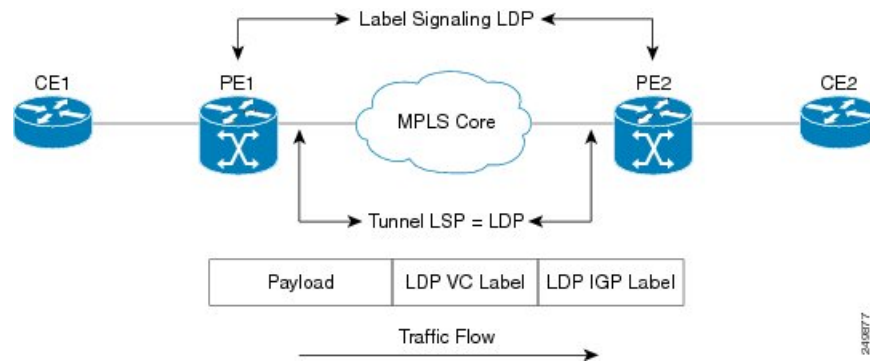
BGP-based VPLS Autodiscovery with LDP Signaling

A PE router advertises an identifier through BGP for each VPLS instance. This identifier is unique within the VPLS instance and acts like a VPLS ID. The identifier enables the PE router receiving the BGP advertisement to identify the VPLS associated with the advertisement and import it to the correct VPLS instance. In this manner, for each VPLS, a PE router learns the other PE routers that are members of the VPLS.

The signaling of pseudowires between provider edge devices, uses targeted LDP sessions to exchange label values and attributes. Forwarding Equivalence Class (FEC) 129 is used for the signaling. The information carried by FEC 129 includes the VPLS ID, the Target Attachment Individual Identifier (TAII) and the Source Attachment Individual Identifier (SAII).

The LDP advertisement also contains the inner label or VPLS label that is expected for the incoming traffic over the pseudowire. This enables the LDP peer to identify the VPLS instance with which the pseudowire is to be associated and the label value that it is expected to use when sending traffic on that pseudowire.

Figure 7: Discovery and Signaling Attributes



Configuring BGP-based VPLS Autodiscovery with LDP Signaling

BGP and LDP need to be configured as indicated in the section [Configuring BGP and LDP for BGP-based Autodiscovery, on page 62](#) before proceeding to the configurations in this section.

Configuration Example:

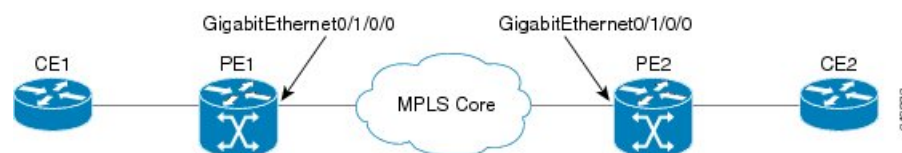
The below code block shows the basic configuration steps required for BGP-based VPLS autodiscovery with LDP Signaling.

```
Router(config)# l2vpn
Router(config-l2vpn)# bridge group {bridge group name}
Router(config-l2vpn-bg)# bridge-domain {bridge domain name}
Router(config-l2vpn-bg-bd)# vfi {vfi name}
Router(config-l2vpn-bg-bd-vfi)# autodiscovery bgp
Router(config-l2vpn-bg-bd-vfi-ad)# vpn-id 10
Router(config-l2vpn-bg-bd-vfi-ad)# rd auto
Router(config-l2vpn-bg-bd-vfi-ad)# route-target 1.1.1.1:100
Router(config-l2vpn-bg-bd-vfi-ad)# signaling-protocol ldp
Router(config-l2vpn-bg-bd-vfi-ad-sig)# vpls-id 120:200
Router(config-l2vpn-bg-bd-vfi-ad-sig)# commit
```

Running Configuration:

The following figure illustrates an example of configuring VPLS with BGP autodiscovery (AD) and LDP Signaling.

Figure 8: VPLS with BGP autodiscovery and LDP signaling



Configuration at PE1:

```

l2vpn
  router-id 10.10.10.10
  bridge group bg1
  bridge-domain bd1
  vfi vf1
  vpn-id 100
  autodiscovery bgp
  rd 1:100
  router-target 12:12
  signaling-protocol ldp
  vpls-id 120:200

```

Configuration at PE2:

```

l2vpn
  router-id 20.20.20.20
  bridge group bg1
  bridge-domain bd1
  vfi vf1
  vpn-id 100
  autodiscovery bgp
  rd 2:200
  router-target 12:12
  signaling-protocol ldp
  vpls-id 120:100

```

BGP-based VPWS Autodiscovery

BGP-based autodiscovery is possible even for point-to-point L2VPN services such as VPWS. However, true autodiscovery is not possible in VPWS as it is in VPLS. In VPWS, in order to connect the Customer Edge (CE) routers, an explicit configuration has to be done at each PE. Only the existence of other PEs can be indicated by autodiscovery.

BGP-based VPWS Autodiscovery with BGP Signaling

The two primary functions of the VPWS control plane are: auto-discovery and signaling. Both of these functions are accomplished with a single BGP Update advertisement.

When a VPWS cross-connect is configured with BGP autodiscovery and signaling enabled, BGP distributes NLRI for the cross-connect with the PE as the BGP next-hop and appropriate CE-ID. Additionally, the cross-connect is associated with one or more BGP export Route Targets (RTs) that are also distributed (along with NLRI).

Configuring BGP-based VPWS Autodiscovery with BGP Signaling

BGP and LDP need to be configured as indicated in the section [Configuring BGP and LDP for BGP-based Autodiscovery, on page 62](#) before proceeding to the configurations in this section.

Configuration Example:

The below code block shows the basic configuration steps required for BGP-based VPWS autodiscovery with BGP Signaling.

```

Router(config)# l2vpn
Router(config-l2vpn)# xconnect group {xconnect group name}
Router(config-l2vpn-xc)# mp2mp {instance name}
Router(config-l2vpn-xc-mp2mp)# vpn-id {vpn-id}
Router(config-l2vpn-xc-mp2mp)# l2-encapsulation vlan
Router(config-l2vpn-xc-mp2mp)# autodiscovery bgp
Router(config-l2vpn-xc-mp2mp-ad)# rd auto
Router(config-l2vpn-xc-mp2mp-ad)# route-target 2.2.2.2:100
Router(config-l2vpn-xc-mp2mp-ad)# signaling-protocol bgp
Router(config-l2vpn-xc-mp2mp-ad-sig)# ce-id 1
Router(config-l2vpn-xc-mp2mp-ad-sig-ce)# interface GigabitEthernet0/1/0/1.1 remote-ce-id 2

```

Running Configuration:

The following figure illustrates an example of configuring VPWS with BGP autodiscovery and BGP Signaling.

Figure 9: VPWS with BGP autodiscovery and BGP signaling



Configuration at PE1:

```

l2vpn
xconnect group gr1
mp2mp mp1
vpn-id 100
l2 encapsulation vlan
autodiscovery bgp
rd auto
route-target 2.2.2.2:100
! Signaling attributes
signaling-protocol bgp
ce-id 1
interface GigabitEthernet0/1/0/1.1 remote-ce-id 2

```

Configuration at PE2:

```

l2vpn
xconnect group gr1
mp2mp mp1
vpn-id 100
l2 encapsulation vlan
autodiscovery bgp
rd auto
route-target 2.2.2.2:100
! Signaling attributes
signaling-protocol bgp
ce-id 2
interface GigabitEthernet0/1/0/2.1 remote-ce-id 1

```

Verification:

PE1:

```
PE1# show l2vpn discovery xconnect
```

```
Service Type: VPWS, Connected
```

```
List of VPNs (1 VPNs):
```

```
XC Group: gr1, MP2MP mp1
```

```
List of Local Edges (1 Edges):
```

```
Local Edge ID: 1, Label Blocks (1 Blocks)
```

Label base Offset	Size	Time Created
16030	10	01/24/2009 21:23:04

```
Status Vector: 9f ff
```

```
List of Remote Edges (1 Edges):
```

```
Remote Edge ID: 2, NLRIs (1 NLRIs)
```

Label base Offset	Size	Peer ID	Time Created
16045	10	1.1.1.1	01/24/2009 21:29:35

```
Status Vector: 7f ff
```

```
PE1# show l2vpn xconnect mp2mp detail
```

```
Group gr1, MP2MP mp1, state: up
```

```
VPN ID: 100
```

```
VPN MTU: 1500
```

```
L2 Encapsulation: VLAN
```

```
Auto Discovery: BGP, state is Advertised (Service Connected)
```

```
Route Distinguisher: (auto) 3.3.3.3:32770
```

```
Import Route Targets:
```

```
2.2.2.2:100
```

```
Export Route Targets:
```

```
2.2.2.2:100
```

```
Signaling protocol: BGP
```

```
CE Range: 10
```

```
...
```

```
Group gr1, XC mp1.1:2, state is up; Interworking none
```

```
Local CE ID: 1, Remote CE ID: 2, Discovery State: Advertised
```

```

AC: GigabitEthernet0/1/0/1.1, state is up
  Type VLAN; Num Ranges: 1
  VLAN ranges: [1, 1]
  MTU 1500; XC ID 0x2000013; interworking none
PW: neighbor 1.1.1.1, PW ID 65538, state is up ( established )
  PW class not set, XC ID 0x2000013
  Encapsulation MPLS, Auto-discovered (BGP), protocol BGP
    MPLS          Local          Remote
    -----
    Label          16031          16045
    MTU            1500          1500
    Control word enabled          enabled
    PW type        Ethernet VLAN    Ethernet VLAN
    CE-ID          1            2
    -----
...
PE1# show bgp l2vpn vpws
BGP router identifier 3.3.3.3, local AS number 100
BGP generic scan interval 60 secs
BGP table state: Active
Table ID: 0x0
BGP main routing table version 913
BGP NSR converge version 3
BGP NSR converged
BGP scan interval 60 secs
Status codes: s suppressed, d damped, h history, * valid, > best
                i - internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
  Network          Next Hop      Rcvd Label    Local Label
Route Distinguisher: 1.1.1.1:32775
*>i2:1/32          1.1.1.1      16045         nolabel

```

```
*>i3:1/32          1.1.1.1          16060          no-label
Route Distinguisher: 3.3.3.3:32770 (default for vrf gr1:mp1)
*> 1:1/32          0.0.0.0          no-label          16030
*>i2:1/32          1.1.1.1          16045          no-label
*>i3:1/32          1.1.1.1          16060          no-label
```

Processed 5 prefixes, 5 paths

PE2:

```
PE2# show l2vpn discovery xconnect
```

Service Type: VPWS, Connected

List of VPNs (1 VPNs):

XC Group: gr1, MP2MP mp1

List of Local Edges (2 Edges):

Local Edge ID: 2, Label Blocks (1 Blocks)

Label base	Offset	Size	Time Created
16045	1	10	01/24/2009 21:09:14

Status Vector: 7f ff

Local Edge ID: 3, Label Blocks (1 Blocks)

Label base	Offset	Size	Time Created
16060	1	10	01/24/2009 21:09:14

Status Vector: 7f ff

List of Remote Edges (1 Edges):

Remote Edge ID: 1, NLRIs (1 NLRIs)

Label base	Offset	Size	Peer ID	Time Created
16030	1	10	3.3.3.3	01/24/2009 21:09:16

Status Vector: 9f ff

```
PE2# show l2vpn xconnect mp2mp detail
```

Group gr1, MP2MP mp1, state: up


```

VPN ID: 100

VPN MTU: 1500

L2 Encapsulation: VLAN

Auto Discovery: BGP, state is Advertised (Service Connected)
    Route Distinguisher: (auto) 1.1.1.1:32775
    Import Route Targets:
        2.2.2.2:100
    Export Route Targets:
        2.2.2.2:100
    Signaling protocol:BGP
    CE Range:10

...

Group grp1, XC mpl.2:1, state is up; Interworking none

Local CE ID: 2, Remote CE ID: 1, Discovery State: Advertised

AC: GigabitEthernet0/1/0/2.1, state is up

    Type VLAN; Num Ranges: 1

    VLAN ranges: [1, 1]

    MTU 1500; XC ID 0x2000008; interworking none

PW: neighbor 3.3.3.3, PW ID 131073, state is up ( established )

    PW class not set, XC ID 0x2000008

Encapsulation MPLS, Auto-discovered (BGP), protocol BGP

    MPLS          Local          Remote
    -----
    Label          16045          16031
    MTU            1500          1500
    Control word   enabled        enabled
    PW type        Ethernet VLAN  Ethernet VLAN
    CE-ID          2              1
    -----

...

PE2# show bgp l2vpn vpws

```

```

BGP router identifier 1.1.1.1, local AS number 100

BGP generic scan interval 60 secs

BGP table state: Active

Table ID: 0x0

BGP main routing table version 819

BGP NSR converge version 7

BGP NSR converged

BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
                i - internal, S stale

Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop      Rcvd Label    Local Label
Route Distinguisher: 1.1.1.1:32775 (default for vrf gr1:mp1)
*>i1:1/32           3.3.3.3       16030         noLabel
*> 2:1/32           0.0.0.0       noLabel       16045
*> 3:1/32           0.0.0.0       noLabel       16060

Route Distinguisher: 3.3.3.3:32770
*>i1:1/32           3.3.3.3       16030         noLabel

Processed 4 prefixes, 4 paths

```

BGP-based VPWS Autodiscovery with LDP Signaling

Signaling of pseudowires requires exchange of information between two endpoints. LDP is better suited for point-to-point signaling.

A PE router advertises an identifier through BGP for the VPWS instance. The signaling of pseudowires between provider edge devices uses targeted LDP sessions to exchange label values and attributes and to configure the pseudowires. FEC 129 is used for the signaling. The information carried by FEC 129 includes the **xconnect** ID, the Target Attachment Individual Identifier (TAII) and the Source Attachment Individual Identifier (SAII).

Configuring BGP-based VPWS Autodiscovery with LDP Signaling

BGP and LDP need to be configured as indicated in the section [Configuring BGP and LDP for BGP-based Autodiscovery, on page 62](#) before proceeding to the configurations in this section.

Configuration Example:

The below code block shows the basic configuration steps required for BGP based VPWS autodiscovery with LDP Signaling.

```
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group {xconnect group name}
Router(config-l2vpn-xc)# mp2mp {instance name}
Router(config-l2vpn-xc-mp2mp)# vpn-id {vpn-id}
Router(config-l2vpn-xc-mp2mp)# l2-encapsulation vlan
Router(config-l2vpn-xc-mp2mp)# autodiscovery bgp
Router(config-l2vpn-xc-mp2mp-ad)# rd auto
Router(config-l2vpn-xc-mp2mp-ad)# route-target 2.2.2.2:100
Router(config-l2vpn-xc-mp2mp-ad)# signaling-protocol ldp
Router(config-l2vpn-xc-mp2mp-ad-sig)# ce-id 1
Router(config-l2vpn-xc-mp2mp-ad-sig-ce)# interface GigabitEthernet0/1/0/1.1 remote-ce-id 2
```

Running Configuration:

The following figure illustrates an example of configuring VPWS with BGP autodiscovery and LDP Signaling.

Figure 10: VPWS with BGP autodiscovery and LDP signaling

**Configuration at PE1:**

```
l2vpn
xconnect group gr1
mp2mp mp1
vpn-id 100
l2 encapsulation vlan
autodiscovery bgp
rd auto
route-target 2.2.2.2:100
! Signaling attributes
signaling-protocol ldp
ce-id 1
interface GigabitEthernet0/1/0/1.1 remote-ce-id 2
```

Configuration at PE2:

```
l2vpn
xconnect group gr1
mp2mp mp1
vpn-id 100
l2 encapsulation vlan
autodiscovery bgp
rd auto
route-target 2.2.2.2:100
! Signaling attributes
signaling-protocol ldp
ce-id 2
interface GigabitEthernet0/1/0/2.1 remote-ce-id 1
```




CHAPTER 7

Storm Control

Storm Control provides Layer 2 port security under a Virtual Private LAN Services (VPLS) bridge by preventing excess traffic from disrupting the bridge. This module describes how to configure traffic storm control.

- [Storm Control](#), on page 75
- [Supported Traffic Types for Storm Control](#), on page 76
- [Storm Control Thresholds](#), on page 76
- [Restrictions](#), on page 76
- [Configure Storm Control](#), on page 76
- [Related Topics](#), on page 77
- [Associated Commands](#), on page 77

Storm Control

A traffic storm occurs when packets flood a VPLS bridge, creating excessive traffic and degrading network performance. Storm control prevents VPLS bridge disruption by suppressing traffic when the number of packets reaches configured threshold levels. You can configure separate threshold levels for different types of traffic on an access circuit (AC) under a VPLS bridge.

Storm control monitors incoming traffic levels on a port and drops traffic when the number of packets reaches the configured threshold level during any 1-second interval. The 1-second interval is set in the hardware and is not configurable. On Cisco NCS 5000 Series Router, the monitoring interval is always one second. The number of packets allowed to pass during this interval is configurable, per port, per traffic type. During this interval, it compares the traffic level with the storm control level that the customer configures. When the incoming traffic reaches the storm control level configured on the bridge port, storm control drops traffic until the end of storm control interval. At the beginning of a new interval, traffic of the specified type is allowed to pass on the port. The thresholds are configured using a packets-per-second (pps) and kilobit-per-second (kbps) rate.

Storm control has little impact on router performance. Packets passing through ports are counted regardless of whether the feature is enabled. Additional counting occurs only for the drop counters, which monitor dropped packets. Storm control counts the number of packets dropped per port. The drop counters are cumulative for all traffic types.

Supported Traffic Types for Storm Control

On each VPLS bridge port, you can configure up to three storm control thresholds—one for each of the supported traffic types. If you do not configure a threshold for a traffic type, then storm control is not enabled on that port or interface for that traffic type.

The supported traffic types are:

- Broadcast traffic—Packets with a packet destination MAC address equal to FFFF.FFFF.FFFF.
- Multicast traffic—Packets with a packet destination MAC address not equal to the broadcast address, but with the multicast bit set to 1. The multicast bit is bit 0 of the most significant byte of the MAC address.
- Unknown unicast traffic—Packets with a packet destination MAC address not yet learned.

Storm Control Thresholds

Storm control thresholds are configured at a packet-per-second and kilobit-per-second rate. A threshold is the number of packets of the specified traffic type that can pass on a port during a 1-second interval. Valid values for storm control thresholds are integers from 1 to 160000. Only kbps rate is supported by hardware. However, pps is supported; pps rate is converted to kbps. The pps rate is calculated as 1 pps = 8 kbps.

Restrictions

- Storm control parameters must be configured only at the bridge-domain level. Interface or AC level configurations are not supported.
- Storm control rates can be applied at the physical port level and not at the sub-interface level. Hence, the same storm control rates are applied to the sub-interfaces on the given physical port.
- Storm control is not supported for forwarding pseudowires (VFI PWs).
- No alarms are generated when packets are dropped.
- Only kbps rate is supported by hardware. Though the pps configuration is allowed, it is converted to kbps. The pps rate is calculated as 1 pps = 8 kbps.

Configure Storm Control

The storm control feature is disabled by default. It must be explicitly enabled on each port or bridge-domain for each traffic type. The thresholds are configured using a packets-per-second (pps) or kilobit-per-second (kbps) rate. Perform this task to configure storm control on an access circuit (AC).

Configuration Example

```
RP/0/RSP0/CPU0:router# configure
```

```
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group csco
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet0/1/0/0.100
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# storm-control broadcast kbps 4500
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# commit
```

Running Configuration

```
configure
l2vpn
bridge group csco
bridge-domain abc
interface GigabitEthernet0/1/0/0.100
storm-control broadcast kbps 4500
!
```

Related Topics

- [Storm Control, on page 75](#)
- [Supported Traffic Types for Storm Control, on page 76](#)

Associated Commands

- storm-control

Related Topics

- [Storm Control, on page 75](#)
- [Supported Traffic Types for Storm Control, on page 76](#)

Associated Commands

- storm-control



CHAPTER 8

Configure Multiple Spanning Tree Protocol

This chapter introduces you to Multiple Spanning Tree Protocol (MSTP) which is one of the variants of Spanning Tree Protocol (STP) and describes how you can configure the MSTP feature.

- [Overview of Spanning Tree Protocol, on page 79](#)
- [Overview of MSTP, on page 80](#)
- [MSTP Support on Cisco NCS 5000 Series Routers, on page 80](#)
- [Configuring MSTP, on page 82](#)
- [Configuring MSTP BPDU Guard, on page 84](#)
- [References for Spanning Tree Protocol, on page 86](#)

Overview of Spanning Tree Protocol

Ethernet is no longer just a link-layer technology used to interconnect network vehicles and hosts. Its low cost and wide spectrum of bandwidth capabilities coupled with a simple plug and play provisioning philosophy have transformed Ethernet into a legitimate technique for building networks, particularly in the access and aggregation regions of service provider networks.

Ethernet networks lacking a TTL field in the Layer 2 (L2) header and, encouraging or requiring multicast traffic network-wide, are susceptible to broadcast storms if loops are introduced. However, loops are a desirable property as they provide redundant paths. Spanning tree protocols (STP) are used to provide a loop free topology within Ethernet networks, allowing redundancy within the network to deal with link failures.

There are many variants of STP; however, they work on the same basic principle. Within a network that may contain loops, a sufficient number of interfaces are disabled by STP so as to ensure that there is a loop-free spanning tree, that is, there is exactly one path between any two devices in the network. If there is a fault in the network that affects one of the active links, the protocol recalculates the spanning tree so as to ensure that all devices continue to be reachable. STP is transparent to end stations which cannot detect whether they are connected to a single LAN segment or to a switched LAN containing multiple segments and using STP to ensure there are no loops.

For more information, see [References for Spanning Tree Protocol, on page 86](#)

Restrictions for STP on Cisco NCS 5000 Series Routers

The following restrictions are applicable for STP on Cisco NCS 5000 Series Routers

- The only type of STP that is supported on Cisco NCS 5000 Series Routers is Multiple Spanning Tree Protocol (MSTP).
- Per vlan Spanning Tree(PVST/PVST+/PVRST) is not supported on Cisco NCS 5000 Series Routers.
- Access gateway feature is not supported.

Overview of MSTP

The Multiple Spanning Tree Protocol (MSTP) is an STP variant that allows multiple and independent spanning trees to be created over the same physical network. The parameters for each spanning tree can be configured separately, so as to cause a different network devices to be selected as the root bridge or different paths to be selected to form the loop-free topology. Consequently, a given physical interface can be blocked for some of the spanning trees and unblocked for others.

Having set up multiple spanning tree instances, the set of VLANs in use can be partitioned among them; for example, VLANs 1 - 100 can be assigned to spanning tree instance 1, VLANs 101 - 200 can be assigned to spanning tree instance 2, VLANs 201 - 300 can be assigned to spanning tree instance 3, and so on. Since each spanning tree has a different active topology with different active links, this has the effect of dividing the data traffic among the available redundant links based on the VLAN—a form of load balancing.

MSTP Support on Cisco NCS 5000 Series Routers

Cisco NCS 5000 Series Routers support MSTP, as defined in IEEE 802.1Q-2005, on physical Ethernet interfaces and Ethernet Bundle interfaces.

In addition, the below Cisco features are supported:

- BPDU Guard—This Cisco feature protects against misconfiguration of edge ports.
- Flush Containment—This Cisco feature helps prevent unnecessary MAC flushes that would otherwise occur following a topology change.
- Bringup Delay—This Cisco feature prevents an interface from being added to the active topology before it is ready to forward traffic.

MSTP BPDU Guard

The MSTP BPDU Guard feature protects against misconfiguration of edge ports.



Note In order to enable the MSTP BPDU Guard feature for an interface, the command **portfast bpduguard** must be configured on it.

Port Fast

The Port Fast feature manage the ports at the edge of the switched Ethernet network. For devices that only have one link to the switched network (typically host devices), there is no need to run MSTP, as there is only

one available path. Furthermore, it is undesirable to trigger topology changes (and resultant MAC flushes) when the single link fails or is restored, as there is no alternative path.

By default, MSTP monitors ports where no BPDUs are received, and after a timeout, places them into edge mode whereby they do not participate in MSTP. When **portfast** is explicitly configured on an interface, MSTP considers that interface to be an edge port and removes it from consideration when calculating the spanning tree. And hence the convergence time for the whole network is improved when **portfast** is configured.

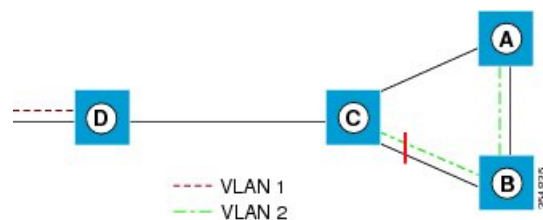


Note MSTP BPDU Guard feature is supported by configuring interfaces in port fast mode. BPDU guard feature will error-disable the port on receiving BPDU packets.

Flush Containment

Flush containment is a Cisco feature that helps prevent unnecessary MAC flushes due to unrelated topology changes in other areas of a network. This is best illustrated by example. The following figure shows a network containing four devices. Two VLANs are in use: VLAN 1 is only used on device D, while VLAN 2 spans devices A, B and C. The two VLANs are in the same spanning tree instance, but do not share any links.

Figure 11: Flush Containment



If the link AB goes down, then in normal operation, as C brings up its blocked port, it sends out a topology change notification on all other interfaces, including towards D. This causes a MAC flush to occur for VLAN 1, even though the topology change which has taken place only affects VLAN 2.

Flush containment helps deal with this problem by preventing topology change notifications from being sent on interfaces on which no VLANs are configured for the MSTI in question. In the example network this would mean no topology change notifications would be sent from C to D, and the MAC flushes which take place would be confined to the right hand side of the network.



Note Flush containment is enabled by default, but can be disabled by configuration, thus restoring the behavior described in the IEEE 802.1Q standard.

Bringup Delay

Bringup delay is a Cisco feature that stops MSTP from considering an interface when calculating the spanning tree, if the interface is not yet ready to forward traffic. This is useful when a line card first boots up, as the system may declare that the interfaces on that card are *Up* before the dataplane is fully ready to forward traffic. According to the standard, MSTP considers the interfaces as soon as they are declared *Up*, and this may cause it to move other interfaces into the blocking state if the new interfaces are selected instead.

Bringup delay solves this problem by adding a configurable delay period which occurs as interfaces that are configured with MSTP first come into existence. Until this delay period ends, the interfaces remain in blocking state, and are not considered when calculating the spanning tree.

Bringup delay only takes place when interfaces which are already configured with MSTP are created, for example, on a card reload. No delay takes place if an interface which already exists is later configured with MSTP.

Configuring MSTP

The different steps involved in configuring MSTP are as follows:

1. Configure VLAN interfaces

```
Router# configure
Router(config)# interface TenGigE0/0/0/2.1001 l2transport
Router(config-subif)# encapsulation dot1q 1001
Router(config)# interface TenGigE0/0/0/3.1001 l2transport
Router(config-subif)# encapsulation dot1q 1001
Router(config)# interface TenGigE0/0/0/14.1001 l2transport
Router(config-subif)# encapsulation dot1q 1001
Router(config)# interface TenGigE0/0/0/2.1021 l2transport
Router(config-subif)# encapsulation dot1q 1021
Router(config)# interface TenGigE0/0/0/3.1021 l2transport
Router(config-subif)# encapsulation dot1q 1021
Router(config)# interface TenGigE0/0/0/14.1021 l2transport
Router(config-subif)# encapsulation dot1q 1021
Router(config-subif)# commit
```

2. Configure L2VPN bridge-domains with the VLAN interfaces configured in the previous step.

```
Router# configure
Router(config)# l2vpn bridge group mstp
Router(config-l2vpn-bg)# bridge-domain mstp1001
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/2.1001
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/3.1001
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/14.1001
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# exit
Router(config-l2vpn-bg)# exit
Router(config-l2vpn-bg)# bridge-domain mstp1021
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/2.1021
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/3.1021
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/14.1021
Router(config-l2vpn-bg-bd-ac)# commit
```

3. Configure MSTP.

```
Router# configure
Router(config)# spanning-tree mst abc
Router(config-mstp)# name mstp1
```

```

Router(config-mstp)# instance 1001
Router(config-mstp-inst)# vlan-ids 1001-1020
Router(config-mstp-inst)# exit
Router(config-mstp)# instance 1021
Router(config-mstp-inst)# vlan-ids 1021-1040
Router(config-mstp-inst)# exit
Router(config-mstp)# int tenGigE 0/0/0/2
Router(config-mstp-if)# exit
Router(config-mstp)# int tenGigE 0/0/0/3
Router(config-mstp-if)# exit
Router(config-mstp)# int tenGigE 0/0/0/14
Router(config-mstp-if)# commit

```

Running Configuration for MSTP

```

!
Configure
/* Configure VLAN interfaces */
interface TenGigE0/0/0/2.1001 l2transport
 encapsulation dot1q 1001
!
interface TenGigE0/0/0/3.1001 l2transport
 encapsulation dot1q 1001
!
interface TenGigE0/0/0/14.1001 l2transport
 encapsulation dot1q 1001

interface TenGigE0/0/0/2.1021 l2transport
 encapsulation dot1q 1021
!
interface TenGigE0/0/0/3.1021
 l2transport
 encapsulation dot1q 1021
!
interface TenGigE0/0/0/14.1021 l2transport
 encapsulation dot1q 1021
!
/* Configure L2VPN Bridge-domains */
l2vpn
 bridge group mstp
  bridge-domain mstp1001
  interface TenGigE0/0/0/2.1001
  !
  interface TenGigE0/0/0/3.1001
  !
  interface TenGigE0/0/0/14.1001
  !
 bridge-domain mstp1021
  interface TenGigE0/0/0/2.1021
  !
  interface TenGigE0/0/0/3.1021
  !
  interface TenGigE0/0/0/14.1021
!
/* Configure MSTP */
spanning-tree mst abc
 name mstp1
 instance 1001
  vlan-ids 1001-1020
!

```

```

instance 1021
  vlan-ids 1021-1040
  !
interface TenGigE0/0/0/2
  !
interface TenGigE0/0/0/3
  !
interface TenGigE0/0/0/14

```

Verification for MSTP

The MSTP configuration can be verified using the command **show spanning-tree mst**

```

/* Verify the MSTP configuration */
Router# show spanning-tree mst abc instance 121
Mon Jan 23 12:11:48.591 UTC
Role: ROOT=Root, DSGN=Designated, ALT=Alternate, BKP=Backup, MSTR=Master
State: FWD=Forwarding, LRN=Learning, BLK=Blocked, DLY=Bringup Delayed

```

Operating in dot1q mode

MSTI 121:

VLANS Mapped: 121-130

```

Root ID      Priority      32768
Address     dceb.9456.b9d4
This bridge is the root
Int Cost    0
Max Age 20 sec, Forward Delay 15 sec

```

```

Bridge ID   Priority      32768 (priority 32768 sys-id-ext 0)
Address     dceb.9456.b9d4
Max Age 20 sec, Forward Delay 15 sec
Max Hops 20, Transmit Hold count 6

```

Interface	Port ID Pri.Nbr Cost	Role State	Designated Bridge ID	Port ID Pri.Nbr
BE1	128.1 10000	DSGN FWD	32768 dceb.9456.b9d4	128.1
Te0/0/0/1	128.2 2000	DSGN FWD	32768 dceb.9456.b9d4	128.2
Te0/0/0/16	128.3 2000	DSGN FWD	32768 dceb.9456.b9d4	128.3
Te0/0/0/17	128.4 2000	DSGN FWD	32768 dceb.9456.b9d4	128.4

Configuring MSTP BPDU Guard

This section describes how you can configure MSTP BPDU Guard.

```

Router# configure
Router(config)# l2vpn bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd1
Router(config-l2vpn-bg-bd)# int TenGigE 0/0/0/7
Router(config-l2vpn-bg-bd-ac)# root

```

```

Router(config)# spanning-tree mst m0
Router(config-mstp)# interface tenGigE 0/0/0/7
Router(config-mstp-if)# portfast bpduguard
Router(config-mstp-if)# root
Router(config)# int tenGigE 0/0/0/7 l2transport
Router(config-if-l2)# commit

```

Running Configuration with MSTP BPDU Guard

```

!
Configure
l2vpn
  bridge group bg1
  bridge-domain bd1
  interface TenGigE0/0/0/7
  !
spanning-tree mst m0
  interface TenGigE0/0/0/7
  portfast bpduguard
!
interface TenGigE0/0/0/7
  l2transport
!

```

Verification for MSTP BPDU Guard

Verify that you have configured MSTP BPDU Guard.

```

/* Verify the MSTP BPDU Guard configuration */
Router# show interfaces tenGigE 0/0/0/7
Wed Nov  9 09:23:56.268 UTC
TenGigE0/0/0/7 is error disabled, line protocol is administratively down
Interface state transitions: 2
Hardware is TenGigE, address is 7cad.7425.c8c8 (bia 7cad.7425.c8c8)
Layer 2 Transport Mode
MTU 1514 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
  reliability 255/255, txload 0/255, rxload 0/255
Encapsulation ARPA,
Full-duplex, 10000Mb/s, link type is force-up
output flow control is off, input flow control is off
Carrier delay (up) is 10 msec
loopback not set,
Last link flapped 00:00:49
Last input 00:00:40, output 00:00:40
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
 38752 packets input, 4611429 bytes, 0 total input drops
 1 drops for unrecognized upper-level protocol
Received 1 broadcast packets, 38751 multicast packets
 0 runts, 0 giants, 0 throttles, 0 parity
 0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort

```

References for Spanning Tree Protocol

This section provides references for STP. For an overview of STP, see [Overview of Spanning Tree Protocol](#), on page 79

STP Operation

All variants of STP operate in a similar fashion: STP frames (known as bridge protocol data units (BPDUs)) are exchanged at regular intervals over Layer 2 LAN segments, between network devices participating in STP. Such network devices do not forward these frames, but use the information to construct a loop free spanning tree.

The spanning tree is constructed by first selecting a device which is the *root* of the spanning tree (known as the root bridge), and then by determining a loop free path from the *root bridge* to every other device in the network. Redundant paths are disabled by setting the appropriate ports into a blocked state, where STP frames can still be exchanged but data traffic is never forwarded. If a network segment fails and a redundant path exists, the STP protocol recalculates the spanning tree topology and activates the redundant path, by unblocking the appropriate ports.

The selection of the root bridge within a STP network is determined by the lowest Bridge ID which is a combination of configured bridge priority and embedded mac address of each device. The device with the lowest priority, or with equal lowest priority but the lowest MAC address is selected as the root bridge.

The selection of the active path among a set of redundant paths is determined primarily by the port path cost. The port path cost represents the cost of transiting between that port and the root bridge - the further the port is from the root bridge, the higher the cost. The cost is incremented for each link in the path, by an amount that is (by default) dependent on the media speed. Where two paths from a given LAN segment have an equal cost, the selection is further determined by the lowest bridge ID of the attached devices, and in the case of two attachments to the same device, by the configured port priority and port ID of the neighboring attached ports.

Once the active paths have been selected, any ports that do not form part of the active topology are moved to the blocking state.

Topology Changes

Network devices in a switched LAN perform MAC learning; that is, they use received data traffic to associate unicast MAC addresses with the interface out of which frames destined for that MAC address should be sent. If STP is used, then a recalculation of the spanning tree (for example, following a failure in the network) can invalidate this learned information. The protocol therefore includes a mechanism to notify topology changes around the network, so that the stale information can be removed (flushed) and new information can be learned based on the new topology.

A *Topology Change* notification is sent whenever STP moves a port from the blocking state to the forwarding state. When it is received, the receiving device flushes the MAC learning entries for all ports that are not blocked other than the one where the notification was received, and also sends its own topology change notification out of those ports. In this way, it is guaranteed that stale information is removed from all the devices in the network.

Variants of STP

There are many variants of the Spanning Tree Protocol:

- **Legacy STP (STP)**—The original STP protocol was defined in IEEE 802.1D-1998. This creates a single spanning tree which is used for all VLANs and most of the convergence is timer-based.
- **Rapid STP (RSTP)**—This is an enhancement defined in IEEE 802.1D-2004 to provide more event-based, and hence faster, convergence. However, it still creates a single spanning tree for all VLANs.
- **Multiple STP (MSTP)**—A further enhancement was defined in IEEE 802.1Q-2005. This allows multiple spanning tree instances to be created over the same physical topology. By assigning different VLANs to the different spanning tree instances, data traffic can be load-balanced over different physical links. The number of different spanning tree instances that can be created is restricted to a much smaller number than the number of possible VLANs; however, multiple VLANs can be assigned to the same spanning tree instance. The BPDUs used to exchange MSTP information are always sent untagged; the VLAN and spanning tree instance data is encoded inside the BPDU.
- **Per-Vlan STP (PVST)**—This is an alternative mechanism for creating multiple spanning trees; it was developed by Cisco before the standardization of MSTP. Using PVST, a separate spanning tree is created for each VLAN. There are two variants: PVST+ (based on legacy STP), and PVRST (based on RSTP). At a packet level, the separation of the spanning trees is achieved by sending standard STP or RSTP BPDUs, tagged with the appropriate VLAN tag.
- **Per-Vlan Rapid Spanning Tree (PVRST)**— This feature is the IEEE 802.1w (RSTP) standard implemented per VLAN, and is also known as Rapid PVST or PVST+. A single instance of STP runs on each configured VLAN (if you do not manually disable STP). Each Rapid PVST+ instance on a VLAN has a single root switch. You can enable and disable STP on a per-VLAN basis when you are running Rapid PVST+. PVRST uses point-to-point wiring to provide rapid convergence of the spanning tree. The spanning tree reconfiguration can occur in less than one second with PVRST (in contrast to 50 seconds with the default settings in the 802.1D STP).
- **Resilient Ethernet Protocol (REP)**— This is a Cisco-proprietary protocol for providing resiliency in rings. It is included for completeness, as it provides MSTP compatibility mode, using which, it interoperates with an MSTP peer.



CHAPTER 9

References

This section provides additional information on understanding and implementing Layer 2 VPNs.

- [Gigabit Ethernet Protocol Standards, on page 89](#)
- [Carrier Ethernet Model References, on page 89](#)
- [Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet, on page 91](#)
- [References for Configuring Link Bundles, on page 92](#)

Gigabit Ethernet Protocol Standards

The 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 the point-of-presence (POP), including core and edge routers and L2 and Layer 3 (L3) switches.

The Gigabit Ethernet interfaces in Cisco NCS 5000 Series Routers support these standards:

- Protocol standards:
 - IEEE 802.3 Physical Ethernet Infrastructure
 - IEEE 802.3ae 10 Gbps Ethernet
- Ethernet standards
 - Ethernet II framing also known as DIX
 - IEEE 802.3 framing also includes LLC and LLC/SNAP protocol frame formats
 - IEEE 802.1q VLAN tagging
 - IEEE 802.1ad Provider Bridges

For more information, see [Carrier Ethernet Model References, on page 89](#).

Carrier Ethernet Model References

This topic covers the references for Gigabit Ethernet Protocol Standards.

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 802.3ae 10 Gbps Ethernet

Under the International Standards Organization's Open Systems Interconnection (OSI) model, Ethernet is fundamentally a L2 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.

General Ethernet Standards

- IEEE 802.1q VLAN tagging—This standard defines VLAN tagging, and also the traditional VLAN trunking between switches. Technically, it also defines QinQ tagging, and MSTP. Cisco NCS 5000 Series Routers do NOT support ISL.
- IEEE 802.1ad Provider Bridges—This standard is a subset of 802.1q and is often referred to as 802.1ad. Cisco NCS 5000 Series Routers do not adhere to the entire standard, but large portions of the standard's functionality are supported.

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 NCS 5000 Series Routers support 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 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. By default, both ingress and egress flow control are off on Cisco NCS 5000 Series Routers.

Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet

The below table 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 by entering the **no shutdown** command.

Table 5: Gigabit Ethernet and 10-Gigabit Ethernet Modular Services Card Default Configuration Values

Parameter	Configuration File Entry	Default Value	Restrictions
Flow control	flow-control	egress on ingress off	none
MTU	mtu	1514 bytes for normal frames 1518 bytes for 802.1Q tagged frames 1522 bytes for QinQ frames	none
MAC address	mac address	Hardware burned-in address (BIA ²)	L3 only
L2 port	l2transport	off/L3	L2 subinterfaces must have L3 main parent interface
Egress filtering	Ethernet egress-filter	off	none
Link negotiation	negotiation	off	physical main interfaces only
Tunneling Ethertype	tunneling ethertype	0X8100	configured on main interface only; applied to subinterfaces only

Parameter	Configuration File Entry	Default Value	Restrictions
VLAN tag matching	encapsulation	all frames for main interface; only ones specified for subinterfaces	encapsulation command only subinterfaces

1. The restrictions are applicable to L2 main interface, L2 subinterface, L3 main interface, interflex L2 interface etc.
2. burned-in address

References for Configuring Link Bundles

This section provides references to configuring link bundles. For an overview of link bundles and configurations, see [Configure Link Bundles for Layer 2 VPNs, on page 33](#).

Characteristics of Link Bundles

- Any type of Ethernet interfaces can be bundled, with or without the use of LACP (Link Aggregation Control Protocol).
- 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.
- A bundle can be administratively enabled or disabled.
- Each individual link within a bundle can be administratively enabled or disabled.
- Ethernet link bundles are created in the same way as EtheroKinet channels, where the user enters the same configuration on both end systems.
- The MAC address that is set on the bundle becomes the MAC address of the links within that bundle.
- When LACP configured, each link within a bundle can be configured to allow different keepalive periods on different members.
- Load balancing 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, work independently on each link within a bundle.
- Upper layer protocols, such as routing updates and hello messages, are sent over any member link of an interface bundle.
- Bundled interfaces are point to point.
- A link must be in the UP state before it can be in distributing state in a bundle.
- 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.

Methods of Forming Bundles of Ethernet Interfaces

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.

For each link configured as bundle member, 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 routers 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 It is recommended that you avoid modifying the MAC address, because changes in the MAC address can affect packet forwarding.

- EtherChannel—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.

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 a router, the peer can be either another router or a switch. LACP monitors the operational state of link bundles to ensure these:

- 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.