Cisco Nexus 7000 Series NX-OS MPLS Configuration Guide

December 2016

Cisco Systems, Inc.
www.cisco.com

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

Cisco Nexus 7000 Series NX-OS MPLS Configuration Guide

CONTENTS

FINAL DRAFT - CISCO CONFIDENTIAL

FINAL DRAFT - CISCO CONFIDENTIAL

FINAL DRAFT - CISCO CONFIDENTIAL

FINAL DRAFT - CISCO CONFIDENTIAL

New and Changed Information

This section provides release-specific information for each new and changed feature in the *Cisco Nexus 7000 Series NX-OS MPLS Configuration Guide*.

Table 1 summarizes the new and changed features for the Cisco Nexus 7000 Series NX-OS MPLS Configuration Guide and tells you where they are documented.Yur software release might not support all the features in this document. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release.

Table i-1 New and Changed Information for MPLS

Feature	Description	Changed in Release	Where Documented
MPLS over GRE to MPLS Stitching	Starting from Cisco NX-OS Release 8.4(2), GRE-based Layer 3 VPN interwork with MPLS or IP VPNs is supported on M3-series I/O modules.	8.4(2)	Chapter 23, “Configuring MPLS over GRE”
MPLS over GRE	Starting from Cisco NX-OS Release 8.3(1), MPLS over GRE is supported on M3-Series I/O modules.	8.3(1)	Chapter 23, “Configuring MPLS over GRE”
Ethernet over Multiprotocol Label Switching	Starting from Cisco NX-OS Release 8.2(1), all EoMPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on M3-Series I/O modules.	8.2(1)	Chapter 27, “Configuring Ethernet over MPLS”
Virtual Private LAN Service	Starting from Cisco NX-OS Release 8.2(1), all VPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on M3-Series I/O modules.	8.2(1)	Chapter 29, “Configuring Virtual Private LAN Service”
BGP VPNv4 Multipath	This feature achieves Equal Cost Multi-Path (ECMP) for traffic flowing from an Autonomous System Border Router (ASBR) towards the Provider Edge (PE) device in an Multi-Protocol Label Switching (MPLS) cloud network by using a lower number of prefixes and MPLS labels.	8.0(1)	BGP VPNv4 Multipath
M3 Series module support	MPLS Layer 3 VPNs are supported on M3 Series modules.	7.3(0)DX(1)	Chapter 20, “Guidelines and Limitations for MPLS Layer 3 VPNs”

Table i-1 New and Changed Information for MPLS (continued)

Feature	Description	Changed in Release	Where Documented
MPLS TE event logging	Added support for logging the LSP and FRR events	7.3(0)D1(1)	Chapter 10, "Logging Label Switched Path (LSP) Events" Chapter 10, "Logging Fast Reroute (FRR) Events"
MPLS TE Cost Limit	Added support for configuring cost limit for TE tunnels.	7.3(0)D1(1)	Chapter 10, "Configuring Cost Limit"
InterAS Option B	This feature was enhanced with the InterAS option B and InterAS option B (with RFC 3107) provisions. An IBGP VPNv4/v6 session between DC edge routers and an EBGP VPNv4/v6 session between DC edge routers and WAN routers can be established without a per VRF iBGP session between DC edge routers..	7.2(0)D1(1)	Chapter 24, "InterAS Option B"
Configuring Automatic Bandwidth Adjustment for MPLS TE Tunnels	Added information about automatic bandwidth adjustment for MPLS TE tunnels.	6.2(6)	Chapter 11, "Configuring Automatic Bandwidth Adjustment for MPLS TE Tunnels"
Any Transport over MPLS	Added Any Transport over MPLS (AToM), which accommodates different types of Layer 2 packets, including Ethernet and VLAN, to enable the service provider to transport different types of traffic over the backbone and accommodate all types of customers.	6.2(2)	Chapter 25, "Configuring Any Transport over MPLS"
Any Transport over MPLS Pseudowire provisioning	Added pseudowire provisioning for AToM. This feature enables you to configure static pseudowires in cases where you cannot use directed control protocols, such as the Label Distribution Protocol or Resource Reservation Protocol over traffic-engineered tunnels (RSVP-TE).	6.2(2)	Chapter 26, "Configuring Any Transport over MPLS Pseudowire Provisioning"
Ethernet over Multiprotocol Label Switching	Added Ethernet over Multiprotocol Label Switching (EoMPLS), which is a Virtual Private Wire Service (VPWS) that is used to carry Layer 2 Ethernet frames over an MPLS network. EoMPLS enables service providers to offer emulated Ethernet services over existing MPLS networks.	6.2(2)	Chapter 27, "Configuring Ethernet over MPLS"
EoMPLS Graceful Restart	Added support for a switch that is configured with the Label Distribution Protocol (LDP) Graceful Restart (GR) to assist its neighboring switches recover gracefully from an interruption in service.	6.2(2)	Chapter 28, "Configuring EoMPLS Layer 2 VPN Graceful Restart"

Table i-1 New and Changed Information for MPLS (continued)

Feature	Description	Changed in Release	Where Documented
Layer 2 and Layer 3 load balancing co-existence	Added Layer 3 VPN and Layer 2 VPN forwarding that is performed independently on the switch using two different types of adjacencies. The forwarding is not be impacted by having a different method of load balancing for the Layer 2 VPN.	6.2(2)	Chapter 22, "Configuring MPLS Layer 3 VPN Load Balancing"
MPLS LSP Ping/Traceroute for LDP/TE and LSP Ping for VCCV	Added support for Virtual Circuit Connectivity Verification (VCCV) in Layer 2 VPN Operations, Administration, and Maintenance (OAM).	6.2(2)	Chapter 34, "Verifying Connectivity with MPLS LSP Ping and Traceroute"
MPLS over GRE	Added a mechanism for tunneling Multiprotocol Label Switching (MPLS) packets over a non-MPLS network.	6.2(2)	Chapter 23, "Configuring MPLS over GRE"
OSPF Sham-Link Support for MPLS VPN	Added a sham-link to connect VPN client sites that run Open Shortest Path First (OSPF) and share back door OSPF links in a Multiprotocol Label Switching (MPLS) VPN configuration.	6.2(2)	Chapter 20, "Configuring MPLS Layer 3 VPNs"
Virtual Private LAN Service	Added a point-to-multipoint service between multiple customer sites using a mesh of point-to-point pseudowires over the provider core to emulate a LAN between the sites.	6.2(2)	Chapter 29, "Configuring Virtual Private LAN Service"
VPLS VPN Pseudowire Redundancy	Added support for detecting a failure in the network and rerouting the Layer 2 service to another endpoint that can continue to provide the service.	6.2(2)	Chapter 30, "Configuring Layer 2 VPN Pseudowire Redundancy"
VPLS Dual-Homing with a vPC	Added support for integrating Virtual Private LAN (VPLS) with the virtual port channel (vPC) functionality in active-standby mode.	6.2(2)	Chapter 31, "Configuring Layer 2 VPN VPLS Dual-Homing with a vPC"
MPLS	Added support for M2 Series modules.	6.1(1)	Chapter 1, "Overview"
MPLS static label binding	Changed the maximum value for the MPLS static label range to 471804.	6.1(1)	Chapter 8, "Configuring MPLS Static Label Binding"
MVPNs	Added support for multicast GRE tunnel interfaces for PE-CE routing with MVPN.	6.1(1)	Chapter 32, "Configuring MVPNs"
MPLS	F2 Series modules do not support MPLS.	6.0(1)	Chapter 1, "Overview"
MPLS Layer 3 VPNs	Added matching and setting support for import maps on standard and extended communities for Cisco NX-OS Release 5.2(7) and later 5.2 releases.	5.2(7)	Chapter 20, "Configuring MPLS Layer 3 VPNs"
MPLS Layer 3 VPNs	Removed the MPLS license requirement for the EIGRP site of origin feature.	5.2(5)	Chapter 20, "Configuring MPLS Layer 3 VPNs"

Table i-1 New and Changed Information for MPLS (continued)

Feature	Description	Changed in Release	Where Documented
MVPNs	Added support for multicast GRE tunnel interfaces for PE-CE routing with MVPN.	5.2(4)	Chapter 32, "Configuring MVPNs"
MPLS	MPLS was introduced as a feature of Cisco NX-OS software for Nexus 7000 Series switches.	5.2(1)	Chapter 20, "Configuring MPLS Layer 3 VPNs"

Preface

This preface describes the audience, organization, and conventions of the *Cisco Nexus 7000 Series NX-OS MPLS Configuration Guide*. It also provides information on how to obtain related documentation.

This chapter includes the following sections:

Audience

This publication is for experienced network administrators who configure and maintain Cisco NX-OS devices.

Document Organization

Note As part of our constant endeavor to remodel our documents to meet our customers' requirements, we have modified the manner in which we document configuration tasks. As a result of this, you may find a deviation in the style used to describe these tasks, with the newly included sections of the document following the new format.

This document is organized into the following chapters:

Title	Description
Chapter 1, "Overview"	Provides an overview of the Multiprotocol Label Switching (MPLS) protocol.
Chapter 2, "Configuring the MPLS Feature Set"	Describes how to install and enable the MPLS feature set on Cisco NX-OS devices

Title	Description
Chapter 3, “Configuring the MPLS Label Distribution Protocol”	Provides an overview of the MPLS Label Distribution Protocol (LDP) and describes how to configure it on Cisco NX-OS devices.
Chapter 4, “Configuring MPLS LDP Autoconfiguration”	Describes how to configure MPLS LDP autoconfiguration on Cisco NX-OS devices.
Chapter 5, “Configuring MPLS LDP Session Protection”	Describes how to configure MPLS LDP session protection on Cisco NX-OS devices.
Chapter 6, “Configuring MPLS LDP IGP Synchronization”	Describes how to configure MPLS LDP IGP synchronization on Cisco NX-OS devices.
Chapter 6, “Configuring MPLS LDP Lossless MD5 Session Authentication”	Describes how to configure MPLS LDP lossless MD5 session authentication on Cisco NX-OS devices.
Chapter 7, “Configuring MPLS LDP Label Filtering”	Describes how to configure MPLS LDP label filtering on Cisco NX-OS devices.
Chapter 8, “Configuring MPLS Static Label Binding”	Describes how to configure MPLS static label binding on Cisco NX-OS devices.
Chapter 9, “Configuring MPLS LDP Graceful Restart”	Describes how to configure MPLS LDP graceful restart on Cisco NX-OS devices.
Chapter 10, “Configuring Basic MPLS TE”	Describes how to configure MPLS traffic engineering (TE) on Cisco NX-OS devices.
Chapter 11, “Configuring Automatic Bandwidth Adjustment for MPLS TE Tunnels”	Describes how to configure MPLS Resource Reservation Protocol (RSVP) on Cisco NX-OS devices.
Chapter 12, “Configuring MPLS RSVP TE”	Describes how to configure MPLS Resource Reservation Protocol (RSVP) on Cisco NX-OS devices.
Chapter 13, “Configuring the Path Selection Metric for MPLS TE Tunnels”	Describes how to configure the path selection metric for MPLS TE tunnels on Cisco NX-OS devices.
Chapter 14, “Configuring LSP Attributes for MPLS TE”	Describes how to configure label switched path (LSP) attributes for path options associated with MPLS TE tunnels on Cisco NX-OS devices.
Chapter 15, “Configuring MPLS TE Verbatim Paths”	Describes how to configure an MPLS TE verbatim path on Cisco NX-OS devices.
Chapter 16, “Configuring MPLS TE Forwarding Adjacency”	Describes how to configure MPLS TE forwarding adjacency on Cisco NX-OS devices.
Chapter 18, “Configuring MPLS TE Class-Based Tunnel Selection”	Describes how to configure MPLS TE class-based tunnel selection on Cisco NX-OS devices.
Chapter 17, “Configuring MPLS TE Path Protection”	Describes how to configure MPLS TE path protection on Cisco NX-OS devices.
Chapter 18, “Configuring MPLS TE Fast Reroute Link and Node Protection”	Describes how to configure MPLS TE fast reroute link and node protection on Cisco NX-OS devices.

Title	Description
Chapter 19, “Configuring MPLS Quality of Service”	Describes how to configure MPLS quality of service (QoS) on Cisco NX-OS devices.
Chapter 20, “Configuring MPLS Layer 3 VPNs”	Describes how to configure MPLS Layer 3 virtual private networks (VPNs) on Cisco NX-OS devices.
Chapter 21, “Configuring MPLS Layer 3 VPN Label Allocation”	Describes how to configure label allocation for MPLS Layer 3 VPNs on Cisco NX-OS devices.
Chapter 22, “Configuring MPLS Layer 3 VPN Load Balancing”	Describes how to configure load balancing for MPLS Layer 3 VPNs on Cisco NX-OS devices.
Chapter 32, “Configuring MVPNs”	Describes how to configure multicast VPNs on Cisco NX-OS devices.
Chapter 34, “Verifying Connectivity with MPLS LSP Ping and Traceroute”	Describes how to troubleshoot MPLS connectivity with MPLS ping and traceroute.
Chapter 33, “Configuring MPLS LSP Multipath Tree Trace”	Describes how to troubleshoot MPLS connectivity with the MPLS LSP Multipath Tree Trace feature.
Appendix A, “Configuration Limits for Cisco NX-OS MPLS”	Lists the maximum configuration limits for MPLS.

Document Conventions

Command descriptions use these conventions:

Convention	Description
boldface font	Commands and keywords are in boldface.
italic font	Arguments for which you supply values are in italics.
[ ]	Elements in square brackets are optional.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.

Screen examples use these conventions:

`screen font`	Terminal sessions and information that the switch displays are in screen font.
`boldface screen font`	Information that you must enter is in boldface screen font.
`italic screen font`	Arguments for which you supply values are in italic screen font.
< >	Nonprinting characters, such as passwords, are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

This document uses the following conventions:

Note Means reader *take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Related Documentation

Cisco NX-OS includes the following documents:

Release Notes

Cisco Nexus 7000 Series NX-OS Release Notes, Release 6.x

NX-OS Configuration Guides

Cisco Nexus 2000 Series Fabric Extender Software Configuration Guide

Cisco Nexus 7000 Series NX-OS Configuration Examples

Cisco Nexus 7000 Series NX-OS FabricPath Configuration Guide

Configuring Feature Set for FabricPath

Cisco Nexus 7000 Series NX-OS Fundamentals Configuration Guide

Cisco Nexus 7000 Series NX-OS High Availability and Redundancy Guide

Cisco Nexus 7000 Series NX-OS Interfaces Configuration Guide

Cisco Nexus 7000 Series NX-OS IP SLAs Configuration Guide

Cisco Nexus 7000 Series NX-OS Layer 2 Switching Configuration Guide

Cisco Nexus 7000 Series NX-OS LISP Configuration Guide

Cisco Nexus 7000 Series NX-OS MPLS Configuration Guide

Cisco Nexus 7000 Series NX-OS Multicast Routing Configuration Guide

Cisco Nexus 7000 Series NX-OS OTV Configuration Guide

Cisco Nexus 7000 Series OTV Quick Start Guide

Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide

Cisco Nexus 7000 Series NX-OS SAN Switching Configuration Guide

Cisco Nexus 7000 Series NX-OS Security Configuration Guide

Cisco Nexus 7000 Series NX-OS System Management Configuration Guide

Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide

Cisco Nexus 7000 Series NX-OS Verified Scalability Guide

Cisco Nexus 7000 Series NX-OS Virtual Device Context Configuration Guide

Cisco Nexus 7000 Series NX-OS Virtual Device Context Quick Start

Cisco NX-OS FCoE Configuration Guide for Cisco Nexus 7000 and Cisco MDS 9500

NX-OS Command References

Cisco Nexus 7000 Series NX-OS Command Reference Master Index

Cisco Nexus 7000 Series NX-OS FabricPath Command Reference

Cisco Nexus 7000 Series NX-OS Fundamentals Command Reference

Cisco Nexus 7000 Series NX-OS High Availability Command Reference

Cisco Nexus 7000 Series NX-OS Interfaces Command Reference

Cisco Nexus 7000 Series NX-OS IP SLAs Command Reference

Cisco Nexus 7000 Series NX-OS Layer 2 Switching Command Reference

Cisco Nexus 7000 Series NX-OS LISP Command Reference

Cisco Nexus 7000 Series NX-OS MPLS Command Reference

Cisco Nexus 7000 Series NX-OS Multicast Routing Command Reference

Cisco Nexus 7000 Series NX-OS OTV Command Reference

Cisco Nexus 7000 Series NX-OS Quality of Service Command Reference

Cisco Nexus 7000 Series NX-OS SAN Switching Command Reference

Cisco Nexus 7000 Series NX-OS Security Command Reference

Cisco Nexus 7000 Series NX-OS System Management Command Reference

Cisco Nexus 7000 Series NX-OS Unicast Routing Command Reference

Cisco Nexus 7000 Series NX-OS Virtual Device Context Command Reference

Cisco NX-OS FCoE Command Reference for Cisco Nexus 7000 and Cisco MDS 9500

Other Software Documents

Cisco NX-OS Licensing Guide

Cisco Nexus 7000 Series NX-OS MIB Quick Reference

Cisco Nexus 7000 Series NX-OS Software Upgrade and Downgrade Guide

Cisco NX-OS System Messages Reference

Cisco Nexus 7000 Series NX-OS Troubleshooting Guide

Cisco NX-OS XML Interface User Guide

Documentation Feedback

To provide technical feedback on this document, or to report an error or omission, please send your comments to nexus7k-docfeedback@cisco.com. We appreciate your feedback.

Communications, Services, and Additional Information

- 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.

Overview

This chapter describes the Multiprotocol Label Switching (MPLS) distribution protocol.

This chapter includes the following sections:

Information About MPLS

MPLS is a high-performance packet forwarding technology that integrates the performance and traffic management capabilities of data link layer (Layer 2) switching with the scalability, flexibility, and performance of network-layer (Layer 3) routing. It enables enterprises and service providers to provide differentiated services without sacrificing the existing infrastructure.

MPLS Terminology

The following MPLS terms are used in this document:

- **Multiprotocol Label Switching (MPLS)**—A highly scalable, data-carrying mechanism that is independent of any data link layer protocol, such as Ethernet, ATM, frame relay, or SONET.

.i.definitions:MPLS;
.i.MPLS:definition;

- **Label Distribution Protocol (LDP)**—A mechanism by which two Label Switch Routers (LSR) exchange label mapping information. This protocol is defined by the IETF (RFC 5036).

.i.definitions:LDP;
.i#.label distribution protocol, see LDP;
.i.LDP:definition;

- **Label Edge Router (LER)**—A router that operates at the edges of an MPLS network. An LER determines and applies the appropriate labels and forwards the labeled packets into the MPLS domain.

.i.definitions:LER;
.i#.label edge router, see LER;
.i.LER:definition;

- **Provider Edge (PE)**—The LER that functions as the ingress and/or egress routers to the MPLS domain.

.i.definitions:PE;
.i#.provider edge routers:see PE;
.i#.provider edge routers:also see LER;
.i.PE:definition;

- **Label Forwarding Information Base (LFIB)**—Routing information used to determine the hop-by-hop path through the network.

.i.definitions:LFIB;
.i#.label forwarding information base, see LFIB;
.i.LFIB:definition;

- **Label Switch Router (LSR)**—A router that switches the labels that are used to route packets through an MPLS network.

.i.definitions:LSR;
.i#.label switch router, see LSR;
.i.LSR:definition;

- **Label Switched Path (LSP)**—A route through an MPLS network, defined by a signaling protocol such as LDP or the Border Gateway Protocol (BGP). The path is set up based on criteria in the forwarding equivalence class (FEC).

.i.definitions:LSP;
.i#.label switch path, see LSP;
.i.LSP:definition;

- **Forwarding Equivalence Class (FEC)**—A set of packets with similar characteristics that might be bound to the same MPLS label. An FEC tends to correspond to a label switched path (LSP); however, an LSP might be used for multiple FECs.

.i.definitions:FEC;
.i#.forward equivalence class, see FEC;
.i.FEC:definition;

Benefits of MPLS

MPLS provides the following benefits to enterprise and service provider networks:

- Scalable support for virtual private network (VPN) services in enterprise and service provider networks.

MPLS VPN is highly scalable and can accommodate increasing numbers of sites and customers. MPLS VPN also supports “any-to-any” communication among VPN sites across the enterprise and service provider network. For each MPLS VPN user, the network appears to function as a private IP backbone over which the user can reach other sites within the VPN organization but not the sites of any other VPN organization.

From a user perspective, MPLS VPN greatly simplifies network routing. For example, an MPLS VPN user can employ the backbone as the default route in communicating with all of the other VPN sites.

- Explicit routing capabilities (also called constraint-based routing or traffic engineering) employ constraint-based routing, in which the path for a traffic flow is the shortest path that meets the resource requirements (constraints) of the traffic flow.

.i.MPLS:benefits:traffic engineering;

In MPLS traffic engineering, such factors as bandwidth requirements, media requirements, and the priority of one traffic flow versus another enable the administrator of an enterprise or service provider network to perform the following tasks:

.i.traffic engineering;

  - Control traffic flow in the network
  - Reduce congestion in the network
  - Make best use of network resources

As the network administrator, you can specify the amount of traffic that you expect to flow between various points in the network (establishing a traffic matrix), while relying on the routing system to perform the following tasks:

  - Calculate the best paths for network traffic
  - Set up the explicit paths to carry the traffic

Label Switching Functions

.i.MPLS:functions;

In conventional Layer 3 forwarding mechanisms, as a packet traverses the network, each router extracts all the information relevant to forwarding the packet from the Layer 3 header. The router uses this information as an index for a routing table lookup to determine the next hop for the packet.

.i.routers:packet headers, analyzing;

In the most common case, the only relevant field in the header is the destination address field, but sometimes other header fields might also be relevant. As a result, the header analysis must be done independently at each router through which the packet passes. A complicated table lookup must also be done at each router.

In label switching, MPLS analyzes the Layer 3 header only once. The Layer 3 header is then mapped into a fixed length, unstructured value called a label.

.i.label;
.i.MPLS:label;

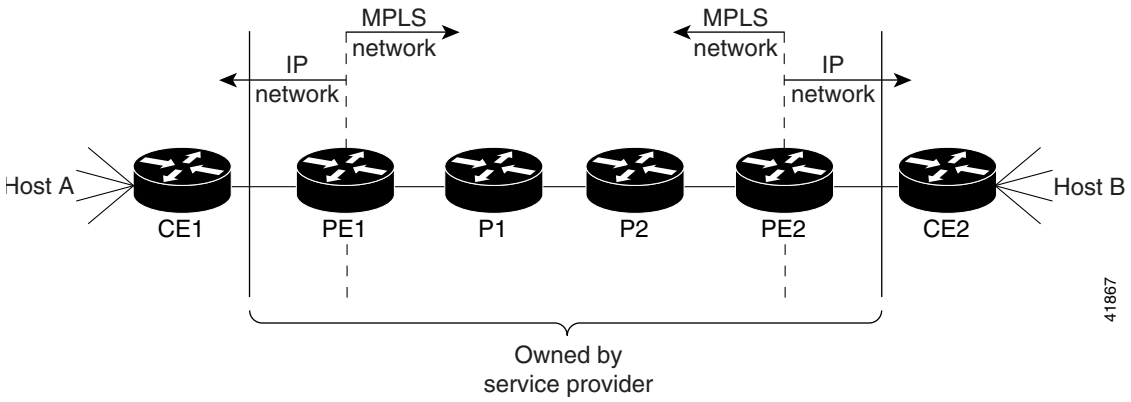
Many different headers can map to the same label, as long as those headers always result in the same choice of the next hop. A label represents a forwarding equivalence class—that is, a set of packets that, however different they may be, are indistinguishable by the forwarding function.

.i.FEC;
.i#.forward equivalence class, see FEC;

The initial choice of a label does not need to be based exclusively on the contents of the Layer 3 packet header; for example, forwarding decisions at subsequent hops can also be based on the routing policy.

Figure 1-1 shows an MPLS network that connects two sites of an IP network that belong to a customer.

Figure 1-1 MPLS Network Connecting Two Sites of a IP Network Belonging to a Customer



Note The network in Figure 1-1 is bidirectional, but in the following discussion, the movement of the packets is from left to right.

Table 1-1 describes the device symbols that are used in Figure 1-1.

Table 1-1 Device Symbols

Symbol	Meaning
CE1	Customer equipment 1
PE1	Service provider edge router (ingress LSR)
P1	Service provider router within the core of the network of the service provider
P2	Service provider router within the core of the network of the service provider
PE2	Service provider edge router (egress LSR)
CE2	Customer equipment 2

Note PE1 and PE2 are at the boundaries between the MPLS network and the IP network.

In Figure 1-1, the following behavior occurs:

1. A packet is sent from CE1 as an IP packet to PE1, which is the provider edge (PE) router.

.i.PE:function;
.i#.provider edge routers:see PE;
.i.ingress label switching router, function;

2. PE1 pushes a label onto the packet—label imposition—and then sends the packet as an MPLS packet to the next hop.

.i.MPLS:packets;

3. The routers P1 and P2 exchange the label on the packet, which is called a label swap, as they transfer it from one machine to the next.
4. PE2 pops the label from the packet, which is called label disposition, and forwards the packet as an IP packet to CE2.

MPLS Label

An MPLS label consists of the following parts:

.i.label stack;
.i.MPLS:label;
.i.MPLS:label stack;

- 20-bit label value.
- 3-bit traffic class field for quality of service (QoS) priority and explicit congestion notification (ECN).
- 1-bit bottom of stack flag. If this flag is set, it signifies that the current label is the last in the stack.
- 8-bit time-to-live (TTL) field.

More than one label can be pushed onto a packet, which is called a label stack. The label stack is inserted between the frame header and the Layer 3 header in the packet.

This section includes the following topics:

- Label Imposition, page 1-10
- Label Swap, page 1-11
- Label Disposition, page 1-12

Label Imposition

.i.label imposition;
.i.MPLS:label imposition;

On the ingress LSR at the provider edge (PE), the incoming packet header is inspected and assigned a label stack that maps it to a particular FEC. The label is pushed onto the packet that is then forwarded to the first hop.

There are different cases for label imposition depending on the configuration, label distribution method, and incoming packet type:

- An incoming IPv4 packet sent to an LDP has an LDP label pushed onto the packet header.
- An incoming IPv4 packet sent to a TE tunnel has a TE label pushed onto the packet header.
- An incoming IPv4 packet sent to a TE tunnel with a backup route has a label stack with a TE backup inner label and a TE backup outer label pushed onto the packet header.
- An incoming IPv4 packet sent to an LDP over a TE tunnel has a label stack with an LDP label and a TE label pushed onto the packet header.
- An incoming IPv4 packet sent to an LDP over a TE tunnel with a backup route has a label stack with an LDP label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.
- An incoming IPv4 packet sent to a Layer 3 VPN in an LDP has a label stack with a VPN label and an LDP label pushed onto the packet header.

- An incoming IPv4 packet sent to a Layer 3 VPN in a TE tunnel has a label stack with a VPN label and a TE label pushed onto the packet header.
- An incoming IPv4 packet sent to a Layer 3 VPN in a TE tunnel with a backup route has a label stack with a VPN label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.
- An incoming IPv4 packet sent to a Layer 3 VPN in an LDP over a TE tunnel has a label stack with a VPN label, an LDP label, and a TE label pushed onto the packet header.
- An incoming IPv4 packet sent to a Layer 3 VPN in an LDP over a TE tunnel with a backup route has a label stack with a VPN label, an LDP label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.

Transporting IPv6 packets over an MPLS backbone is called 6PE/6VPE, where there is no addition of IPv4 headers to the packet:

- An incoming 6PE packet sent to an LDP has a label stack with a BGP label and an LDP label pushed onto the packet header.
- An incoming 6PE packet sent to a TE tunnel has a label stack with a BGP label and a TE label pushed onto the packet header.
- An incoming 6PE packet sent to a TE tunnel with a backup route has a label stack with a BGP label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.
- An incoming 6PE packet sent to an LDP over a TE tunnel has a label stack with a BGP label, an LDP label, and a TE label pushed onto the packet header.
- An incoming 6PE packet sent to an LDP over a TE tunnel with a backup route has a label stack with a BGP label, an LDP label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.
- An incoming 6VPE packet sent to an LDP has a label stack with a VPN label and an LDP label pushed onto the packet header.
- An incoming 6VPE packet sent to a TE tunnel has a label stack with a VPN label and a TE label pushed onto the packet header.
- An incoming 6VPE packet sent to a TE tunnel with a backup route has a label stack with a VPN label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.
- An incoming 6VPE packet sent to an LDP over a TE tunnel has a label stack with a VPN label, an LDP label, and a TE label pushed onto the packet header.
- An incoming 6VPE packet sent to an LDP over a TE tunnel with a backup route has a label stack with a VPN label, an LDP label, a TE backup inner label, and a TE backup outer label pushed onto the packet header.

Label Swap

.i.label swap;
.i.MPLS:label swap;

As the labeled packet traverses the MPLS domain, the outermost label of the label stack is examined at each hop. Depending on the contents of the label, a swap, push (impose), or pop (dispose) operation is performed on the label stack. Forwarding decisions are made by performing a MPLS table lookup for the label carried in the packet header. The packet header does not need to be reevaluated during packet transit through the network. Because the label has a fixed length and is unstructured, the MPLS forwarding table lookup process is both straightforward and fast.

In a swap operation, the label is swapped with a new label, and the packet is forwarded to the next hop that is determined by the new label.

In a push operation, a new label is pushed on top of the existing label, effectively encapsulating the packet in another layer. This process allows hierarchical routing of MPLS packets. Encapsulation is the process used by MPLS VPNs.

In certain cases, the label is swapped and a further label is pushed onto the packet header as follows:

- A packet that traverses a TE tunnel with a backup route has its TE label removed and a label stack with a TE backup inner label and a TE backup outer label are pushed onto the packet header.
- A packet that traverses an LDP over a TE tunnel has its original LDP label removed and a label stack, a new LDP label, and a TE label are pushed onto the packet header.
- A packet that traverses an LDP over a TE tunnel with a backup route has its original LDP label removed and a label stack, a new LDP label, a TE backup inner label, and a TE backup outer label are pushed onto the packet header.

In a pop operation, the label is removed from the packet, which may reveal an inner label below. If the popped label was the last label on the label stack, the packet exits the MPLS domain. Typically, this process occurs at the egress LSR.

Label Disposition

.i.label disposition;
.i.MPLS:label disposition;

On the egress LSR at the provider edge (PE), the MPLS label stack is popped off the packet header leaving an IPv4 or IPv6 packet to be forwarded onward. This process is called disposition.

In certain cases, the MPLS label stack is popped off the packet header at the hop before the egress LSR. This process is called Penultimate Hop Popping (PHP). By using PHP, transit routers that are connected directly to the egress LSR can effectively offload the CPU load on that router by popping the last label themselves and forwarding the packet.

.i#.penultimate hop popping, see PHP;
.i.PHP;

Distribution of Label Bindings

.i.label binding:distribution;

Each LSR in the network makes an independent, local decision as to which label value to use to represent a forwarding equivalence class. This association is known as a *label binding*. Each LSR informs its neighbors of the label bindings it has made. This awareness of label bindings by neighboring routers is facilitated by the following protocols:

.i.label binding;

- LDP—Supports MPLS forwarding along normally routed paths
- Resource Reservation Protocol (RSVP)—Supports MPLS traffic engineering

.i#.resource reservation protocol, see RSVP;
.i.RSVP:traffic engineering;
.i.traffic engineering:RSVP;

- Border Gateway Protocol (BGP)—Supports MPLS VPNs and 6PE/6VPE encapsulation

When a labeled packet is being sent from LSR A to the neighboring LSR B, the label value that is carried by the IP packet is the label value that LSR B assigned to represent the forwarding equivalence class of the packet. The label value changes as the IP packet traverses the network.

MPLS and Routing

.i.label switching:routing;

A label represents a forwarding equivalence class, but it does not represent a particular path through the network. The path through the network continues to be chosen by the existing Layer 3 routing algorithms such as Open Shortest Path First (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), and BGP. That is, at each hop when a label is looked up, the dynamic routing algorithm chooses the next hop.

6PE and 6VPE

You can implement IPv6 on the provider edge (PE) routers over MPLS, which is known as 6PE, and IPv6 VPNs over MPLS, which is known as 6VPE.

IPv6 over MPLS backbones enable isolated IPv6 domains to communicate with each other over an MPLS IPv4 core network. 6PE supports transporting IPv6 traffic over an existing MPLS IPv4 core network. This implementation requires no reconfiguration of core routers because forwarding is based on labels rather than on the IP header itself, which provides a cost-effective strategy for deploying IPv6.

6PE relies on multiprotocol BGP extensions in the IPv4 network configuration on the PE router to exchange IPv6 reachability information in addition to an MPLS label for each IPv6 address prefix to be advertised. PE routers are configured as dual stacks, running both IPv4 and IPv6, and use the IPv4 mapped IPv6 address for IPv6 prefix reachability exchange. The next hop advertised by the PE router for 6PE/6VPE prefixes is still the IPv4 address that is used for IP v4 L3 VPN routes. A value of ::FFFF: is prepended to the IPv4 next hop, which is an IPv4-mapped IPv6 address.

You use dual-stack PE routers, running both IPv4 and IPv6 and an IPv4-mapped IPv6 address for the next hop when exchanging IPv6-prefix reachability information. The system uses multiprotocol BGP (MP-BGP) with labels to exchange IPv6 routes and sets up an MPLS LSP between two PE routers that use IPv4 routing and signaling. The ingress PE router imposes the BGP label and directs IPv6 traffic into the LSP based on the IP-mapped IPv6 next hop. Again, the core routers use switch labels; they do not do any IPv6 forwarding. The egress PE router forwards the IPv6 packet based on the inner label or by performing a route lookup.

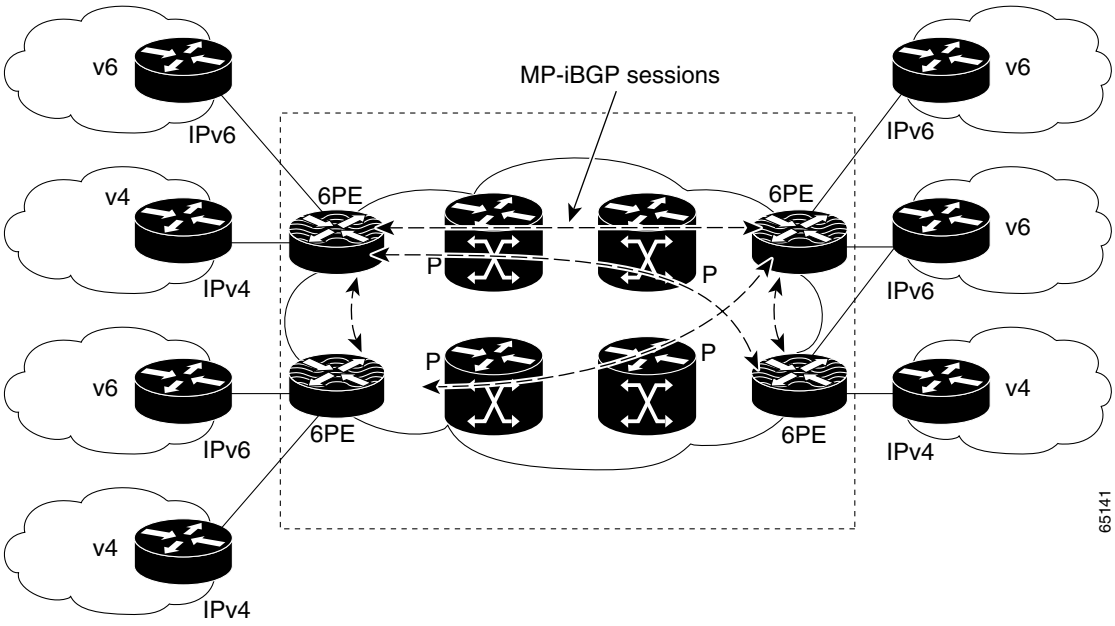
The system imposes a hierarchy of labels on the 6PE ingress router to keep the IPv6 traffic transparent to all the core routers. The bottom label, which is automatically assigned to the IPv6 prefix of the destination, is distributed by multiprotocol BGP and used at each 6PE egress router for IPv6 forwarding.

Internal and external BGP multipath for IPv6 allows the IPv6 router to load balance between several paths—for example, the same neighboring autonomous system (AS) or the sub-AS as the same metric—to reach its destination. The 6PE multipath feature uses multiprotocol internal BGP (MP-iBGP) to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route. When you enable multipath on the 6PE router by entering the **maximum-paths** command, you install all labeled paths in the forwarding table with MPLS information (label stack) when MPLS information is available. This functionality enables 6PE and 6VPE to perform load balancing.

Note You must configure all participating iBGP peers with the **address-family ipv6 labeled-unicast** command.

In Figure 1-2, the 6PE routers are configured as dual-stack routers that can route both IPv4 and IPv6 traffic. Each 6PE router is configured to run a protocol to bind the IPv4 labels. The 6PE routers use MP-iBGP to exchange the reachability information with the other 6PE devices within the MPLS domain and to distribute aggregate IPv6 labels between them. All 6PE and core routers (labeled P routers in Figure 1-2) within the MPLS domain share a common IPv4 Interior Gateway Protocol (IGP) such as OSPF or Intermediate System-to-Intermediate System (IS-IS).

Figure 1-2 6PE Router Topology



In addition to the regular MPLS commands for troubleshooting, enter the **show bgp ipv6** and **show ipv6 route** commands.

MPLS Forwarding with 6VPE

6VPE supports VPN connectivity over an MPLS IPv4 provider core network. This feature is very similar to 6PE, but the main difference is that the system uses VRF tables for forwarding lookups at the PE and uses VPN address-families in BGP.

Upon receiving IPv6 traffic from one customer site, the ingress PE router uses MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE router identified as the BGP next hop. The ingress PE router typically prepends the IPv6 packets with the outer and inner labels before putting the packet on the egress interface. At the MPLS penultimate hop label popping, the remaining BGP label identifies the egress PE interface toward the customer site. It also hides the protocol version (IPv6) from the last P router, which would otherwise need to forward an IPv6 packet. A P router is ignorant about IPv6 VPN routes. The IPv6 header remains hidden under one or more MPLS labels.

You can use the **ping6** and **traceroute6** commands to validate data-plane connectivity and to detect any blackholing of traffic. In addition, you can use the **show forwarding ipv6 route** command and regular MPLS commands for troubleshooting.

External and Internal Border Gateway Protocol (EIBGP) is supported for 6VPE, and it functions like the equivalent IPv4 L3 VPN feature.

See Part 5 of this guide for more information about Layer 3 VPNs and 6VPEs.

MPLS Label Switching and HA

.i.label switching:high availability;
The Cisco NX-OS architecture and high availability (HA) infrastructure provide support for feature components to be restarted and resume operations transparently to other services on the device and on neighboring devices. This feature allows for continuous operation with no data loss during planned software changes and unplanned software failures.

MPLS Label Switching supports these Cisco NX-OS HA features:

- Nonstop forwarding (NSF)
- Stateful HA

MPLS Label Switching supports these Cisco NX-OS HA technologies to allow NSF and stateful HA:

- Stateful process restart
- Stateful switchover (SSO)
- In-Service Software Upgrade (ISSU)

Virtualization Support for MPLS

.i.label switching:virtualization support;
The software supports virtual device contexts (VDCs). MPLS configuration and operations are local to the VDC.

Note See the *Cisco Nexus 7000 Series NX-OS Virtual Device Context Configuration Guide* for complete information on VDCs and assigning resources.

Guidelines and Limitations for MPLS

MPLS has the following guidelines and limitations:

- CE devices installs type 3 LSAs with DN-bit or Type 5 LSAs with DN-bit and VPN Route TAG in the RIB (non-default VRF). This behaviour is applicable prior to Cisco NX-OS Release 8.3(2).
- The **default-information originate** command must be configured so that the MPLS default route is advertised to the CE-VRF. When using default-information originate command, the DN-bit in type 3 5 LSAs options and Route TAGs in Type 5 LSAs are not set for the default route only.
- To accommodate the MPLS labels that are pushed onto the packet, you must configure the maximum transmission unit (MTU) for core-facing LDP interfaces to be larger than the default.
- The M1 and M2 Series modules support all Cisco NX-OS MPLS features.

Note F1 Series I/O modules do not support MPLS natively, but they can take advantage of proxy routing with M Series modules for MPLS forwarding. For more information on proxy routing, see the *Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide*.

- F2 Series I/O modules do not support MPLS.
- Before the 6PE or 6VPE features can be implemented, MPLS must be running over the core IPv4 network.
- Dual-stack PE routers are supported but not a required configuration for 6PE.

Updated for CSCvn00190

- When the OTV VDC and the MPLS VDC share the same instance of the M2 forwarding engine (FE), there is a chance for traffic blackholing. The blackholing is because of the MPLS label in MPLS VDC overlap with the MPLS label, which is used to encode the OTV extended VLAN ID (OTV MPLS label = VLAN ID + 32) in the OTV VDC.

  This traffic blackholing problem can be avoided by the following methods:

  – You need to allocate the interfaces on the same M2 FE in such a way that the interfaces are not shared between multiple VDCs that utilize the MPLS.

    For N7K-M224XP-23L (24-port 10GE): ports 1 to 12 are served by FE 0, and ports 13 to 24 are served by FE 1.

    For N7K-M206FQ-23L (6-port 10/40GE): ports 1 to 3 are served by FE 0, and ports 4 to 6 are served by FE 1.

  – Configure the **mpls label range** <*lowest*> <*highest*> command in the MPLS VDC to exclude all labels that can be used for OTV VLAN transport (top of the range is 4094 + 32 = 4196) from the dynamic allocation. For example: mpls label range4127 1028093

Note You need to reload the MPLS VDC to reallocate the existing labels within this range.

- **MPLS Deaggregate Labels Reserve:**

  In NXOS release 8.0(1) a change was introduced that caused deaggregation FECs (aka Per-VRF, or VPN FECs) to have their MPLS labels allocated dynamically from the normal dynamic label range rather than a special reserved block, and the labels were no longer reserved in Resource Manager. This change was introduced in order to support the requirement for more than 500K labels; the maximum configurable value of the dynamic range needed to be increased past the start of the special deaggregation range to allow more labels.

  This caused two problems:

  – In older line cards (M2) a single Forwarding Engine (SoC) instance could have it's ports divided into multiple VDCs. For these line cards the deaggregation label used for each VDC must be unique. This was handled by using the Resource Manager reservations.

  – If a single instance is split among different VDCs, the MPLS labels used by OTV must not overlap with the deaggregation labels.

  As a result, MPLS can blackhole traffic if you have multiple VDCs on the same instance of a M2 module. This issue will not let you upgrade to Cisco NX-OS Release 8.x from an earlier release. M2 module gets affected when ports are shared across VDCs from a same instance and more than one VDC is using MPLS L3VPN, or OTV and L3VPN are both used simultaneously. This is a known limitation. It is recommended that you either revert to a release earlier than Cisco NX-OS Release 8.0(1) or make sure that the M2 instances are not split across VDCs.

  Proposed solution available from Cisco NX-OS Release 8.4(1):

The overall idea for the fix is to make the deaggregation label range default to being reserved in Cisco NX-OS Release 8.4(1), like it was before (prior to 8.0(1)), but allow modification based on the "[no] mpls deaggregate labels reserve" command. The value of the CLI must be the same on all VDCs. For situations where defaulting to the old behaviour may cause traffic loss, the functionality will behave in the new (8.0(1)) manner, but after the next reload reverts to the old behavior (prior to 8.0(1)).

The following explains how the solution works in different scenarios:

- Upgrade from version prior to Cisco NX-OS Release 8.0(1) to the current version through cold boot:

  When the current version boots up it will default to the legacy deaggregation behavior. This happens in all VDCs. Since this is a cold boot, there are no labels allocated.

- Upgrade from version prior to Cisco NX-OS Release 8.0(1) to the current version through ISSU:

  In this scenario first the standby supervisor reloads to the current version, the system switches over to that supervisor, then the new standby reloads with the current version.

  When original standby reloads with the current version while the active is running the old version, it will default to the legacy deaggregation behaviour (prior to Cisco NX-OS Release 8.0(1)). Since the old version must have allocated only deaggregation labels and non-deaggregation labels will operate correctly and the same as in the old version.

- Upgrade from version later than /from the Cisco NX-OS Release 8.0(1) to the current release through cold boot:

  In this scenario when the current version boots up it will default to the legacy deaggregation behaviour. This happens in all VDCs. Since this is a cold boot, there are no labels allocated.

- Upgrade from version later than /from the Cisco NX-OS Release 8.0(1) to the current release through ISSU:

  When the standby supervisor reloads with the current version, it cannot follow the legacy deaggregation behaviour because there may be deaggregation labels outside the deaggregation range and non- deaggregation labels inside the deaggregation range due to the deaggregation range change.

  As the upgrade proceeds the standby supervisor will reload and will perform a check and detect if a label conflict situation is present. If so, it will internally set state so that it will behave as if the "no reserve" command was entered, but this state will not be saved in the config PSS. Similarly, the active supervisor will reload and become standby and perform the same check so that it behaves identically to the old standby.

  Sometime after the ISSU when the switch is reloaded the functionality will default to the "reserve" behaviour and so correct the relevant issues.

- User adds new VDC in current version and enables an MPLS feature in that VDC:

  There are two scenarios here:

a) The other VDCs have "no mpls deaggregate labels reserve". When an MPLS feature is added, the default will be "mpls deaggregate labels reserve", but the VDC conflict issue will still be present because not all VDCs have the reserve version. However, ideally the user will have observed the warning printed either when the "no reserve" command was entered on the other VDC or when the ISSU occurred.

b) The other VDCs have "mpls deaggregate labels reserve". Then the default in the current VDC will match the others and behaviour will be correct.

– Upgrade from version with smaller dynamic range configured that will not contain legacy deaggregation range:

If the user upgrades from a previous version to the current version and has a dynamic range that does not fit completely, it will still allow allocation. For example, if the pre-upgrade dynamic range was set to 1000-2000, after upgrade if a client requests a dynamic label it will work as in pre-Cisco NX-OS Release 8.0(1) images.

If the dynamic range partially overlaps, for example it is 16-500000 (where the LDR is 492287-524286), then only the portion 492287-500000 would be reserved to prevent dynamic allocation by other clients for regular dynamic labels in the deaggregation range. If the dynamic range is later reconfigured the functionality is adjusted appropriately.

– show running config:

If the internal state is set to “reserve” do NOT display “reserve” in show running. This is a default.

If the internal state is set to “no reserve” display “reserve” in show running.

– show running config all:

When “show running config all” is shown it will display the defaults and also the configured values.

If the internal state is set to “reserve” display “reserve”, and if the state is set to “no reserve” display “no reserve”.

Configuring the MPLS Feature Set

This chapter describes how to install and enable the Multiprotocol Label Switching (MPLS) feature set on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About the MPLS Feature Set

MPLS functionality is grouped into a feature set. A feature set is a collection of components that performs a specific set of functions.

You must install and enable the MPLS feature set before you can configure the components that make up the feature set. To do so, follow the instructions in this chapter before configuring the other MPLS components documented in this guide.

Licensing Requirements for the MPLS Feature Set

Product	License Requirement
Cisco NX-OS	MPLS requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for the MPLS Feature Set

The MPLS feature set has the following prerequisites:

- You must install an MPLS license before you can enable and configure any features that belong to the MPLS feature set.

Guidelines and Limitations for the MPLS Feature Set

The MPLS feature set has the following configuration guidelines and limitations:

- You must install the MPLS feature set in the default VDC before you can enable the feature set in any VDC (including the default VDC).
- The MPLS feature set operation might cause the standby supervisor to reload if it is in an unstable state, such as following a service failure or powering up. You can check whether the standby supervisor is stable with the **show system redundancy ha status** command, which shows the high-availability state for all VDCs. When the standby supervisor is stable, it is indicated as “HA standby.”
- Starting from Cisco NX-OS Release 8.0(1), scale support for MPLS labels has been increased. This may lead to dropping of MPLS packets if you configure multiple VDCs that split up a single instance of Enhanced Address Resolution Logic 8 (EARL8) on M2-Series I/O modules. This is applicable for Cisco NX-OS Release 8.0(1) and later releases.

Default Settings for the MPLS Feature Set

Table 2-1 lists the default settings for the MPLS feature set parameters.

Table 2-1 Default MPLS Feature Set Parameters

Parameters	Default
MPLS feature set	Uninstalled and disabled

Configuring the MPLS Feature Set

This section includes the following topics:

Installing the MPLS Feature Set

You must install the MPLS feature set in the default VDC.

Prerequisites

Ensure that you are in the default VDC.

SUMMARY STEPS

1. **configure terminal**
2. **[no] install feature-set mpls**
3. (Optional) **show feature-set**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**[no] install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set in the default VDC. **Note** Use the **no** form of this command to uninstall the MPLS feature set. You cannot uninstall the feature set if it is enabled in any VDC.
Step 3	**show feature-set** **Example:** `switch(config)# show feature-set`	(Optional) Displays the status of the MPLS feature set on the device.

Enabling the MPLS Feature Set

You can enable the installed MPLS feature set in any VDC on the device.

Prerequisites

Ensure that you have installed the MPLS feature set in the default VDC.

Ensure that you are in the correct VDC or use the **switchto vdc** command.

SUMMARY STEPS

1. **configure terminal**
2. **[no] feature-set mpls**
3. (Optional) **show feature-set**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**[no] feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature set. **Note** The feature set must be installed before it shows as an option to this command. **Note** Use the **no** form of this command to disable the MPLS feature set. This command might take some time to complete if the size of the configuration is very large. The command must clean up all of the configurations associated with the MPLS feature set.
Step 3	**show feature-set** **Example:** `switch(config)# show feature-set`	(Optional) Displays the status of the MPLS feature set on the device.

Allowing the MPLS Feature Set in a VDC

By default, the installed MPLS feature set is allowed in all VDCs on the device. You can disallow the installed MPLS feature set in a specific VDC, and you can subsequently allow that disallowed MPLS feature set in the VDC.

Prerequisites

Ensure that you have installed the MPLS feature set in the default VDC.

SUMMARY STEPS

1. **configure terminal**
2. **vdc** *vdc-id*
3. **[no] allow feature-set mpls**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**vdc** *vdc-id* **Example:** switch(config)# vdc 1 switch(config-vdc)#	Specifies a VDC and enters VDC configuration mode.
Step 3	[no] allow feature-set mpls **Example:** switch(config-vdc)# allow feature-set mpls	Allows the MPLS feature set in the VDC. **Note** Use the **no** form of this command to disallow the MPLS feature set in the VDC. You cannot disallow the MPLS feature set if it is enabled in the specified VDC.

Verifying the MPLS Feature Set Configuration

To display the MPLS feature set configuration, perform one of the following tasks:

Command	Purpose
show feature-set	Displays the status of the feature sets on the device.
show feature-set services mpls	Displays the services used by the MPLS feature set.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for the MPLS Feature Set

The following example shows how to install and enable the MPLS feature set in the default VDC on the device:

```
switch# configure terminal
switch(config)# install feature-set mpls
switch(config)# feature-set mpls
switch(config)# exit
switch#
```

The following example shows how to install the MPLS feature set on the device and then switch to a VDC and enable the feature set in that VDC:

```
switch# configure terminal
switch(config)# install feature-set mpls
switch(config)# exit
switch# switchto vdc vdc1
```

```
switch-vdc1# configure terminal
switch-vdc1(config)# feature-set mpls
switch-vdc1(config)# switchback
switch#
```

Note Instead of the **switchto** command, you can use the **ssh** command to connect to the management port of the VDC.

The following example shows how to display the status of the feature sets on the device:

```
switch# show feature-set
Feature Set Name       ID      State
-------------------- -------- ----------
fcoe                    1      installed
fabricpath              2      enabled
fex                     3      disabled
mpls                    4      enabled
switch#
```

The following example shows how to display the services used by the MPLS feature set:

```
switch# show feature-set services mpls
ulib
rsvp
mpls_te
mpls_oam
mpls
ldp
6 services in feature set mpls
switch#
```

Additional References for the MPLS Feature Set

This section includes additional information that is related to the MPLS feature set commands.

Related Documents

Related Topic	Document Title
Cisco NX-OS licensing	*Cisco NX-OS Licensing Guide*
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
VDCs	*Cisco Nexus 7000 Series NX-OS Virtual Device Context Configuration Guide, Release 5.x*
Cisco IOS MPLS LDP	MPLS Label Distribution Protocol (LDP)

Feature History for the MPLS Feature Set

Table 2-2 lists the release history for this feature.

Table 2-2 Feature History for the MPLS Feature Set

Feature Name	Releases	Feature Information
MPLS feature set	5.2(1)	This feature was introduced.

Configuring the MPLS Label Distribution Protocol

This chapter describes how to configure the Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LDP

MPLS LDP enables peer label-switched routers (LSRs) to exchange label binding information for supporting hop-by-hop forwarding in an MPLS network.

The following topics provide information about MPLS LDP:

Introduction to MPLS LDP

MPLS LDP provides the means for LSRs to request, distribute, and release label prefix binding information to peer routers in a network. LDP enables LSRs to discover potential peers and to establish LDP sessions with those peers for the purpose of exchanging label binding information.

MPLS LDP enables one LSR to inform another LSR of the label bindings it has made. Once a pair of routers communicates the LDP parameters, they establish a label-switched path (LSP). MPLS LDP enables LSRs to distribute labels along normally routed paths to support MPLS forwarding. This method of label distribution is also called *hop-by-hop forwarding*. With IP forwarding, when a packet arrives at a router, the router looks at the destination address in the IP header, performs a route lookup, and forwards the packet to the next hop. With MPLS forwarding, when a packet arrives at a router, the router looks at the incoming label, looks up the label in a table, and then forwards the packet to the next hop. MPLS LDP is useful for applications that require hop-by-hop forwarding, such as MPLS virtual private networks (VPNs).

MPLS LDP Functional Overview

MPLS LDP provides the building blocks for MPLS-enabled applications, such as MPLS VPNs.

MPLS LDP provides a standard methodology for hop-by-hop, or dynamic label, distribution in an MPLS network by assigning labels to routes that have been chosen by the underlying Interior Gateway Protocol (IGP) routing protocols. The resulting label-switched paths (LSPs) forward label traffic across an MPLS backbone to particular destinations. These capabilities enable operators to implement MPLS-based IP VPNs.

MPLS LDP Sessions

When you enable MPLS LDP, the LSRs send out messages to try to find other LSRs with which they can create LDP sessions. The following sections explain the differences between directly connected LDP sessions and nondirectly connected LDP sessions.

Directly Connected MPLS LDP Sessions

If an LSR is one hop from its neighbor, it is directly connected to its neighbor. The LSR sends out LDP link hello messages as User Datagram Protocol (UDP) packets to all the routers on the subnet (multicast). A neighboring LSR may respond to the link hello message, allowing the two routers to establish an LDP session. This process is called *basic discovery*.

To initiate an LDP session between routers, the routers determine which router takes the active role and which router takes the passive role. The router that takes the active role establishes an LDP TCP connection and initiates the negotiation of the LDP session parameters. To determine the roles, the two routers compare their transport addresses. The router with the higher IP address takes the active role and establishes the session.

After the LDP TCP connection is established, the LSRs negotiate the session parameters, including the method of label distribution to be used. Two methods are available:

- **Downstream unsolicited**—An LSR advertises label mappings to peers without being asked to.
- **Downstream on demand**—An LSR advertises label mappings to a peer only when the peer asks for them.

Note Downstream on demand is a feature of LDP, but it is not supported in Cisco NX-OS.

Nondirectly Connected MPLS LDP Sessions

If the LSR is more than one hop from its neighbor, it is nondirectly connected to its neighbor. For these nondirectly connected neighbors, the LSR sends out a targeted hello message as a UDP packet that is specifically addressed to that LSR (unicast). The nondirectly connected LSR responds to the hello message, and the two routers begin to establish an LDP session. This process is called *extended discovery*.

An MPLS LDP targeted session is a label distribution session between routers that are not directly connected. You establish nondirectly connected MPLS LDP sessions by enabling the transmission of targeted hello messages.

The exchange of targeted hello messages between two nondirectly connected neighbors can occur in several ways, including the following:

- Router 1 sends targeted hello messages that carry a response request to Router 2. Router 2 sends targeted hello messages in response if its configuration permits. In this situation, Router 1 is considered to be active, and Router 2 is considered to be passive.
- Router 1 and Router 2 both send targeted hello messages to each other. Both routers are considered to be active. Both, one, or neither router can also be passive, if they have been configured to respond to requests for targeted hello messages from each other.

The default behavior of an LSR is to ignore requests from other LSRs that send targeted hello messages. You can configure an LSR to respond to requests for targeted hello messages using the **discovery targeted-hello accept** command.

LDP Label Bindings and Label Spaces

An *LDP label binding* is an association between a destination prefix and a label. The label used in a label binding is allocated from a set of possible labels called a *label space*.

LDP supports two types of label spaces:

- **Interface-specific**—An interface-specific label space uses interface resources for labels. Depending on its configuration, an LDP platform may support zero, one, or more interface-specific label spaces.

- **Platform-wide**—An LDP platform supports a single platform-wide label space for use by interfaces that can share the same labels. For Cisco platforms, all interface types use the platform-wide label space.

LDP Identifiers

LDP uses a 6-byte quantity called an *LDP Identifier* (or *LDP ID*) to name label spaces. The LDP ID is made up of the following components:

- The first four bytes, called the *LDP router ID*, identify the LSR that owns the label space.
- The last two bytes, called the *local label space ID*, identify the label space within the LSR. For the platform-wide label space, the last two bytes of the LDP ID are always both 0.

The LDP ID takes the following form: *LDP router ID* : *local label space ID*.

The following are examples of LDP IDs:

- 172.16.0.0:0
- 192.168.0.0:3

The following steps describe the default process for determining the LDP router ID:

1. The router examines the IP addresses of all operational interfaces.
2. If these IP addresses include loopback interface addresses, the router selects the largest loopback address as the LDP router ID.

   Configuring a loopback address helps to ensure a stable LDP ID for the router because the state of loopback addresses does not change, even in the presence of link-down events. Generally, it is also desirable for the LDP router ID to be preserved across reboots.

   The loopback IP address does not become the LDP router ID under the following circumstances:

   – If the loopback interface has been explicitly shut down.

   – If you configured a different interface to be used as the LDP router ID.

   Note If you use a loopback interface, make sure that the IP address for the loopback interface is configured with a /32 network mask. In addition, make sure that the routing protocol in use is configured to advertise the corresponding /32 network.

3. Otherwise, the router selects the largest IP address that pertains to an operational interface as the LDP router ID.

Note Although the LDP algorithm for selecting a router ID attempts to select a loopback interface, it cannot be guaranteed across all startup scenarios. Therefore, we recommend that you explicitly configure the LDP router ID.

MPLS LDP Transport Address

The default method for determining the LDP router ID might result in a router ID that is not usable in certain situations. For example, the router might select an IP address as the LDP router ID that the routing protocol cannot advertise to a neighboring router. Therefore, you may want to specify the IP

address of an interface as the LDP router ID. When you do so, the router selects the IP address of the specified interface (if the interface is operational) the next time it is necessary to select an LDP router ID, which is typically the next time the current router ID interface is shut down or its address is changed.

The establishment of an LDP session between two routers requires a session TCP connection by which label advertisements can be exchanged between the routers. To establish the session TCP connection, each router must know the transport address (IP address) of the other router.

The LDP discovery mechanism allows a router to advertise the transport address for its end-of-session TCP connection. When the transport address advertisement is explicit, the transport address appears as part of the contents of the discovery hello messages that are sent to the peer. When the transport address advertisement is implicit, the transport address is not included in the discovery hello messages, and the peer uses the source IP address of the received hello messages as the peer transport address.

Note When a router has multiple links that connect it to its peer device, the router must advertise the same transport address in the LDP discovery hello messages that it sends on all such interfaces.

Explicit-Null Labels

Typically, LDP advertises an implicit-null label for directly connected prefixes. The implicit-null label causes the second to last (penultimate) LSR to remove the MPLS header from the packet. In this case, the penultimate LSR and the last LSR do not have access to the quality-of-service (QoS) values that the packet carried before the MPLS header was removed. To preserve the QoS values, you can configure the LSR to advertise an explicit-null label (with a value of zero). The LSR at the penultimate hop forwards MPLS packets with a null label instead of forwarding IP packets.

Note An explicit null label is not needed when the penultimate hop receives MPLS packets with a label stack that contains at least two labels and penultimate hop popping is performed. In this case, the inner label can still carry the QoS value needed by the penultimate and edge LSRs to implement their QoS policy.

High Availability for MPLS LDP

The Cisco NX-OS architecture and high availability infrastructure provide support for feature components to be restarted and resume operations transparently to other services on the device and on neighboring devices. This support allows for continuous operation and minimal data loss during planned software changes and unplanned software failures.

MPLS LDP supports these Cisco NX-OS high availability technologies:

- Nonstop Forwarding (NSF)
- Graceful (stateless) restart
- Stateful Switch Over (SSO)
- In-Service Software Upgrade (ISSU)

Licensing Requirements for MPLS LDP

Product	License Requirement
Cisco NX-OS	MPLS LDP requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LDP

MPLS LDP has the following prerequisites:

- The address reserved intrusion detection system (IDS) check is disabled by default. Do not enable this check (using the **hardware ip verify address reserved** command) if you plan to enable MPLS LDP.

Guidelines and Limitations for MPLS LDP

MPLS LDP has the following configuration guidelines and limitations:

- MPLS LDP does not guarantee that a loopback interface will be selected before a physical interface during a reload. We recommend that you explicitly configure the router ID to ensure that the same router ID is selected across router reloads and configuration copies.
- BGP Labeled Unit stitching with MPLS LDP is not currently supported.
- Cisco NX-OS TCP does not support stateful process restarts for itself or for its clients.
- Cisco NX-OS Release 6.1 introduces support for more than four process instances for OSPFv2 per VDC. However, only the first four configured OSPFv2 instances are supported with MPLS LDP.

Default Settings for MPLS LDP

Table 3-1 lists the default settings for MPLS LDP parameters.

Table 3-1 Default MPLS LDP Parameters

Parameters	Default
Global MPLS LDP	Disabled
MPLS LDP on interfaces	Disabled
Label distribution mode	Independent
Label retention mode	Liberal
Label advertisement	Downstream unsolicited
Basic hello times	15-second hold time, 5-second hello interval
Targeted hello times	90-second hold time, 10-second hello interval
Session times	180-second hold time, 60-second keepalive interval

Table 3-1 Default MPLS LDP Parameters (continued)

Parameters	Default
Initial backoff time	15 seconds
Maximum backoff time	120 seconds
Transport address	LDP router ID

Configuring MPLS LDP

This section includes the following topics:

Enabling MPLS LDP Globally

You can enable MPLS LDP globally on an LSR.

Prerequisites

Ensure that the MPLS feature set is installed in the default VDC and enabled in the VDC for which you are configuring LDP. For more information on the MPLS feature set, see the "Configuring the MPLS Feature Set" chapter.

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Make sure that the address reserved IDS check is disabled.

Note This check should be disabled by default. If it is enabled, use the **no hardware ip verify address reserved** command to disable it.

SUMMARY STEPS

1. **configure terminal**
2. **feature mpls ldp**
3. **(Optional) show running-config mpls ldp**
4. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS LDP feature on the device. **Note** When the MPLS LDP feature is disabled on the device, no LDP commands are available.
Step 3	**show running-config mpls ldp** **Example:** `switch(config)# show running-config mpls ldp`	(Optional) Displays the configuration status of MPLS LDP on the device. **Note** When MPLS LDP is disabled on the device, no LDP information is visible.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Enabling MPLS LDP on an Interface

You can enable MPLS LDP on individual interfaces. To enable LDP, you should configure it globally and on each interface where it is needed.

Note Alternatively, you can globally configure MPLS LDP on every interface associated with a specified Interior Gateway Protocol (IGP) instance using the MPLS LDP autoconfiguration feature. Because you do not have to configure LDP separately on each interface, the autoconfiguration feature makes LDP configuration easier, faster, and error free. For more information, see the "Configuring the MPLS Autoconfiguration" chapter.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that the interface on which you want to enable MPLS LDP is up.

Ensure that an IP address is configured for the interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface slot/port*
3. **mpls ip**
4. **(Optional) show mpls interface detail**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface** *interface slot/port* **Example:** `switch(config)# interface ethernet 2/2` `switch(config-if)#`	Specifies the interface on which you are enabling MPLS LDP and enters interface configuration mode.
Step 3	**mpls ip** **Example:** `switch(config-if)# mpls ip`	Enables MPLS LDP on the specified interface.
Step 4	**show mpls interface detail** **Example:** `switch(config-if)# show mpls interface detail`	(Optional) Displays the configuration status of MPLS LDP on the interface.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Enabling Directly Connected MPLS LDP Sessions

You can configure MPLS LDP sessions between two directly connected routers.

Note Alternatively, you can use the **neighbor** *ip-address* **targeted** command to create a targeted session between directly connected MPLS LSRs when the MPLS label forwarding convergence time is an issue. This command can improve the label convergence time for directly connected neighbor LSRs when the links that directly connect them are down. When the links between the neighbor LSRs are up, both the link and targeted hellos maintain the LDP session. If the links between the neighbor LSRs go down, the targeted hellos maintain the session, which allows the LSRs to retain labels that are learned from each other. When a link that directly connects the LSRs comes back up, the LSRs can immediately reinstall labels for forwarding without having to reestablish their LDP session and exchange labels. See the "Configuring MPLS LDP Session Protection" chapter for information on automatically starting targeted hellos to protect directly connected LDP sessions.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that the interface for which you want to establish an MPLS LDP session is up.

Ensure that an IP address is configured for the interface.

SUMMARY STEPS

1. **configure terminal**

2. **interface** *interface slot/port*
3. **mpls ip**
4. exit
5. mpls ldp configuration
6. **(Optional)** discovery hello {**holdtime** *seconds* | **interval** *seconds*}
7. **(Optional) show mpls ldp discovery** [**detail**]
8. **(Optional) show mpls ldp neighbor** [**detail**]
9. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**interface** *interface slot/port* **Example:** switch(config)# interface ethernet 2/2 switch(config-if)#	Configures an interface and enters interface configuration mode.
Step 3	mpls ip **Example:** switch(config-if)# mpls ip	Configures MPLS hop-by-hop forwarding.
Step 4	exit **Example:** switch(config-if)# exit **switch(config)#**	Exits interface configuration mode.
Step 5	mpls ldp configuration **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.
Step 6	**discovery hello** {**holdtime** *seconds* | **interval** *seconds*} **Example:** switch(config-ldp)# discovery hello holdtime 10	**(Optional)** Configures the hold time or interval for directly connected neighbors. The **holdtime** *seconds* keyword-argument pair defines the period of time that a discovered LDP neighbor is remembered without receipt of an LDP hello message from the neighbor. The range is from 1 to 65535 seconds. The **interval** *seconds* keyword-argument pair defines the period of time between the sending of consecutive hello messages. The range is from 1 to 65535 seconds. **Note** If the interval time is larger than a third of the hold time value, the interval is automatically adjusted to a third of the hold time value.
Step 7	**show mpls ldp discovery** [**detail**] **Example:** switch(config-ldp)# show mpls ldp discovery	(Optional) Verifies that the interface is up and is sending discovery hello messages.
Step 8	**show mpls ldp neighbor** [**detail**] **Example:** switch(config-ldp)# show mpls ldp neighbor	(Optional) Shows that the LDP session between routers was successfully established.
Step 9	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Establishing Nondirectly Connected MPLS LDP Sessions

You can configure nondirectly connected MPLS LDP sessions, which enable you to establish an LDP session between routers that are not directly connected.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that the neighbor IP address is reachable, if you want to set up a targeted session to another LSR.

Note When LDP is enabled on a TE tunnel, LDP uses targeted hellos and sets up a targeted session to the tunnel tailend LSR.

Ensure that the tunnel for which you want to establish an MPLS LDP session is up.

Configure the routers at both ends of the tunnel to be active or enable one router to be passive with the **discovery targeted-hello accept** command. (See Step 4 for information on using this command.)

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. neighbor *ip-address* targeted
4. discovery targeted-hello {accept [**from** *prefix-list*] | **holdtime** *seconds* | **interval** *seconds*}
5. **(Optional) show mpls ldp discovery** [**detail**]
6. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 3	**neighbor** *ip-address* **targeted** **Example:** `switch(config-ldp)# neighbor 192.168.1.1 targeted`	Configures a targeted session to another LSR.
Step 4	**discovery targeted-hello** *{***accept** *[***from** *prefix-list]* | **holdtime** *seconds* | **interval** *seconds}* **Example:** `switch(config-ldp)# discovery targeted-hello accept`	Configures the router to respond to requests for targeted-hello messages from all neighbors or from neighbors specified by the optional prefix list or configures the hold time or interval for neighbors that are not directly connected. The **holdtime** *seconds* keyword-argument pair defines the period of time that a discovered LDP neighbor is remembered without receipt of an LDP hello message from the neighbor. The range is from 1 to 65535 seconds. The **interval** *seconds* keyword-argument pair defines the period of time between the sending of consecutive hello messages. The range is from 1 to 65535 seconds. **Note** If the interval time is larger than a third of the hold time value, the interval is automatically adjusted to a third of the hold time value.
Step 5	**show mpls ldp discovery** [**detail**] **Example:** `switch(config-ldp)# show mpls ldp discovery`	(Optional) Verifies that the interface is up and is sending discovery hello messages.
Step 6	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS LDP Backoff Intervals

You can configure the backoff interval to limit the time it takes to establish an LDP session and to prevent neighbors from overwhelming each other while exchanging parameter information. If you configure a backoff interval and a session setup attempt fails due to an incompatibility between routers, each LSR delays its next attempt (that is, backs off), increasing the delay exponentially with each successive failure until the maximum backoff delay is reached.

Note You should change the settings from the default values only if such settings result in undesirable behavior.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on the device.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. **backoff** *initial-backoff max-backoff*
4. **(Optional) show mpls ldp backoff**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 3	**backoff** *initial-backoff max-backoff* **Example:** `switch(config-ldp)# backoff 30 240`	Configures session setup delay parameters for the LDP backoff mechanism: • The *initial-backoff* argument defines the initial backoff value in seconds. The range is from 5 to 2147483. • The *max-backoff* argument defines the maximum backoff value in seconds. The range is from 5 to 2147483.
Step 4	**show mpls ldp backoff** **Example:** `switch(config-ldp)# show mpls ldp backoff`	(Optional) Displays information about the configured session setup backoff parameters and any potential LDP peers with which session setup attempts are being throttled.
Step 5	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring the MPLS LDP Hold Time

You can configure the MPLS LDP hold time to specify how long an LDP session is maintained in the absence of LDP messages from the session peer.

Note When an LDP session is established between two LSRs, the hold time used for the session is the lower of the values configured on the two LSRs.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on the device.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. **holdtime** {*seconds* | **infinite**}
4. **(Optional) show mpls ldp parameters**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 3	**holdtime** {*seconds* | **infinite**} **Example:** `switch(config-ldp)# holdtime 30`	Specifies how long an LDP session is maintained in the absence of LDP messages from the session peer. The range is from 15 to 65535 seconds.
Step 4	**show mpls ldp parameters** **Example:** `switch(config-ldp)# show mpls ldp parameters`	(Optional) Displays the LDP parameters, including the session hold time.
Step 5	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Specifying the LDP Router ID

You can configure the IP address of an interface as the LDP router ID.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on the interface.

Ensure that the specified interface is operational before assigning it as the LDP router ID.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface slot/port*
3. exit
4. mpls ldp configuration
5. **router-id** *interface number* [**force**]
6. **(Optional) show mpls ldp discovery [detail]**
7. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**interface** *interface slot/port* **Example:** switch(config)# **interface loopback 2** **switch(config-if)#**	Configures an interface and enters interface configuration mode.
Step 3	**exit** **Example:** switch(config-if)# **exit** **switch(config)#**	Exits interface configuration mode.
Step 4	**mpls ldp configuration** **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.

	Command	Purpose
Step 5	**router-id** *interface number* [**force**] **Example:** switch(config-ldp)# **router-id loopback 2**	Specifies the preferred interface for determining the LDP router ID, which is typically determined the next time that the interface is shut down or the address is deconfigured. The **force** keyword enables the router ID to take effect more quickly. However, implementing the router ID depends on the current state of the specified interface. If the interface is up and its IP address is not currently the LDP router ID, the LDP router ID is forcibly changed to the IP address of the interface. This forced change in the LDP router ID tears down any existing LDP sessions, releases label bindings learned from the LDP sessions, and interrupts MPLS forwarding activity associated with the bindings. **Note** Cisco strongly recommends that you explicitly specify the router ID and when doing so that you use the loopback interface. **Note** Beginning with Cisco NX-OS Release 5.2(5), the **force** option is not required, unless there is a requirement to always maintain the same router ID. As long as a loopback interface is specified, it is chosen as the router ID upon system or VDC boot-up. This rule does not apply for the physical interface. For Cisco NX-OS releases prior to 5.2(5), the **force** option is required so that the system boots up with the configured router ID, if it is a loopback or physical interface.
Step 6	**show mpls ldp discovery** [**detail**] **Example:** switch(config-ldp)# show mpls ldp **discovery**	(Optional) Displays the LDP identifier for the local router.
Step 7	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring an MPLS LDP Transport Address

If the LDP router ID is not appropriate for your situation, you can specify the transport address advertised in the LDP discovery hello messages sent on an interface. By default, LDP advertises its LDP router ID as the transport address.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on the interface.

SUMMARY STEPS

1. **configure terminal**

2. **interface** *interface slot/port*
3. **mpls ldp discovery transport-address** {*ip-address* | **interface}**
4. **(Optional) show mpls ldp discovery detail**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface** *interface slot/port* **Example:** `switch(config)# interface ethernet 2/2` `switch(config-if)#`	Configures an interface and enters interface configuration mode.
Step 3	**mpls ldp discovery transport-address** {*ip-address* | **interface**} **Example:** `switch(config-if)# mpls ldp discovery transport-address 209.165.200.225`	Specifies the transport address advertised in the LDP discovery hello messages sent on an interface: • The *ip-address* argument specifies that a specific IP address be advertised as the transport address. • The **interface** keyword specifies that the interface IP address be advertised as the transport address.
Step 4	**show mpls ldp discovery detail** **Example:** `switch(config-if)# show mpls ldp discovery detail`	(Optional) Displays the status of the LDP discovery process, including the transport address.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Preserving QoS Settings with an MPLS LDP Explicit-Null Label

You can configure an LSR to advertise an explicit-null label (with a value of zero) in order to preserve QoS values. An explicit-null label is advertised in place of an implicit-null label for directly connected prefixes.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on the device.

SUMMARY STEPS

1. **configure terminal**

2. mpls ldp configuration
3. **explicit-null** [**for** *prefix-list* | **to** *prefix-list* | **for** *prefix-list* **to** *prefix-list*]
4. **(Optional) show mpls forwarding statistics**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.
Step 3	**explicit-null** [**for** *prefix-list* | **to** *prefix-list* | **for** prefix-list **to** *prefix-list*] **Example:** switch(config-ldp)# **explicit-null**	Advertises an explicit-null label in situations where an implicit-null label would normally be advertised.
Step 4	**show mpls forwarding statistics** **Example:** switch(config-ldp)# **show mpls forwarding statistics**	(Optional) Verifies that MPLS packets are forwarded with an explicit-null label (value of zero).
Step 5	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Shutting Down MPLS LDP Services

You can shut down MPLS LDP services.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that an LDP session is established.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. **shutdown**
4. **(Optional) show mpls ldp neighbor** [*ip-address* | *interface*] [**detail**]

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.
Step 3	**shutdown** **Example:** **switch(config-ldp)# shutdown**	Tears down all LDP sessions, withdraws all outgoing labels from the forwarding plane, and frees all local labels that have been allocated. **Note** The **no shutdown** command reactivates LDP service when LDP is shut down.
Step 4	**show mpls ldp neighbor** [*ip-address* | *interface*] [**detail**] **Example:** **switch(config-ldp)# show mpls ldp neighbor detail**	(Optional) Displays the status of LDP sessions.

Verifying the MPLS LDP Configuration

To display the MPLS LDP configuration, perform one of the following tasks:

Command	Purpose
show mpls interface detail	Displays the configuration status of MPLS LDP on the interface.
show mpls ldp backoff	Displays information about the configured session setup backoff parameters and any potential LDP peers with which session setup attempts are being throttled.
show mpls ldp bindings	Displays the MPLS LDP label information base (LIB).
show mpls ldp capabilities	Displays MPLS LDP capabilities information.
show mpls ldp checkpoint	Displays MPLS LDP checkpoint information.
show mpls ldp discovery [detail]	Displays the status of the LDP discovery process.
show mpls ldp neighbor [*ip-address* | *interface*] [**detail**]	Displays the status of LDP sessions.
show mpls ldp parameters	Displays the current LDP parameters.

Command	Purpose
show mpls switching	Displays the MPLS label switching database. This command can be used to verify the consistency of the LDP database and forwarding information.
show running-config interface [tunnel | tunnel-te] *number*	Displays the running configuration for the tunnel interface.
show running-config mpls ldp	Displays the configuration status of MPLS LDP on the device.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS LDP

This section provides configuration examples for MPLS LDP and includes the following topics:

Examples: Configuring Directly Connected MPLS LDP Sessions

The following example shows how to configure MPLS LDP sessions between two directly connected routers.

Router A Configuration

```
switch(config)# interface ethernet 2/1
switch(config-if)# ip address 10.1.1.1 255.255.255.0
switch(config-if)# mpls ip
switch(config-if)# exit
```

Router B Configuration

```
switch(config)# interface ethernet 2/1
switch(config-if)# ip address 10.1.1.2 255.255.255.0
switch(config-if)# mpls ip
switch(config-if)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# discovery hello interval 5
```

The following example shows how to verify that interface Ethernet 2/1 has been configured to use LDP:

```
switch# show mpls interface detail
Interface Ethernet2/1:
        mpls te lif enabled
        ldp enabled
        MPLS operational
        Label space id 0x10000001
        MPLS sub-layer Ethernet2/1-mpls layer(0x26000003)
```

The following example shows how to verify that the interface is up and is sending LDP discovery hello messages and that the period of time between the sending of consecutive hello messages is 5 seconds:

```
switch# show mpls ldp discovery detail
Local LDP Identifier:
    10.1.1.2:0
    Discovery Sources:
    Interfaces:
        Ethernet2/1 (ldp): xmit
        Enabled: Interface config
            Hello interval: 5000 ms; Transport IP addr: 10.1.1.2
            Clients: IPv4
```

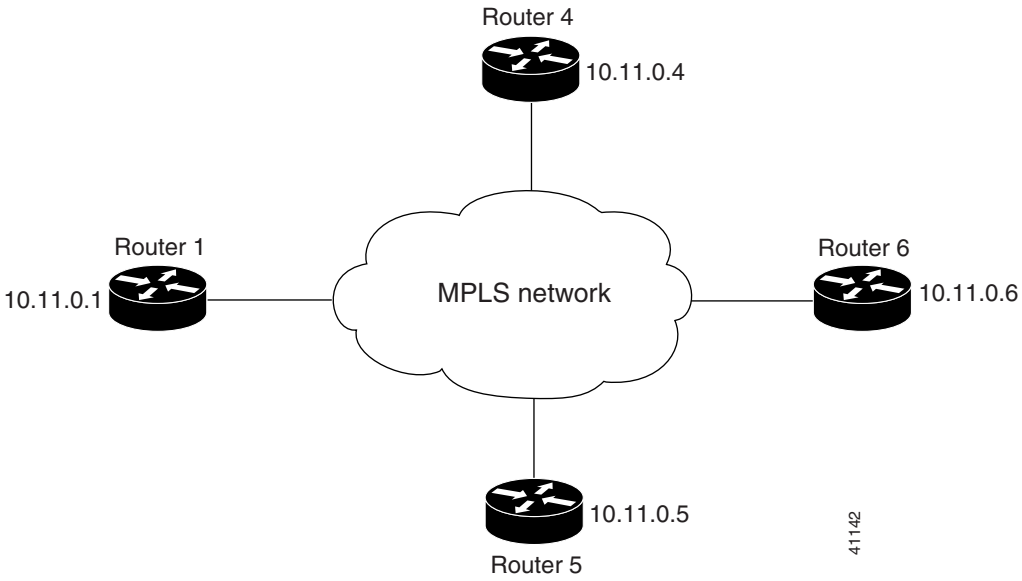
The following example shows how to verify that the LDP session between routers was successfully established:

```
switch# show mpls ldp neighbor
Peer LDP Ident: 10.1.1.1:0; Local LDP Ident 10.1.1.2:0
    TCP connection: 10.1.1.1.646 - 10.1.1.2.12407
    State: Oper; Msgs sent/rcvd: 12/11; Downstream
    Up time: 00:00:10
    LDP discovery sources:
    FastEthernet1/0, Src IP addr: 10.20.10.2
    Addresses bound to peer LDP Ident:
    10.1.1.2    10.20.20.1    10.20.10.2
```

Examples: Establishing Nondirectly Connected MPLS LDP Sessions

The following examples show the configuration of platforms for MPLS LDP nondirectly connected sessions using the sample network shown in Figure 3-1. Note that Routers 1, 4, 5, and 6 in this sample network are not directly connected to each other.

Figure 3-1 Sample Network for Configuring LDP for Targeted Sessions



The configuration example shows the following:

- Targeted sessions between Routers 1 and 4 and Routers 1 and 5 use LDP. Routers 1, 4, and 5 are active.
- Targeted sessions between Routers 1 and 6 use LDP. Router 1 is active and Router 6 is passive.

Router 1 Configuration

```
interface Loopback0            !Loopback interface for LDP ID.
ip address 10.25.0.11 255.255.255.255
```

Router 4 Configuration

```
interface Loopback0            !Loopback interface for LDP ID.
ip address 10.25.0.44 255.255.255.255
```

Router 5 Configuration

```
interface Loopback0            !Loopback interface for LDP ID.
ip address 10.25.0.55 255.255.255.255
```

Router 6 Configuration

By default, a router cannot be a passive neighbor in targeted sessions. Therefore, Router 1, Router 4, and Router 5 are active neighbors in any targeted sessions. The **discovery targeted-hello accept** command permits Router 6 to be a passive target in targeted sessions with Router 1. Router 6 can also be an active neighbor in targeted sessions, although the example does not include such a configuration.

```
interface Loopback0            !Loopback interface for LDP ID.
ip address 10.25.0.66 255.255.255.255

mpls ldp configuration
discovery targeted-hello accept from LDP_SOURCES
                                    !Respond to requests for targeted hellos
                                    !from sources permitted by prefix list LDP_SOURCES

ip prefix-list LDP_SOURCES         !Define prefix list for targeted hello sources
permit 10.11.25.11/32                !Accept targeted hello request from Router 1
deny any                             !Deny requests from other sources
```

Examples: Specifying the LDP Router ID

The following example shows how to assign interface Ethernet 2/2 as the LDP router ID:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# router-id loopback 0 force
```

The following example shows how to display the LDP router ID (10.15.15.15):

```
switch# show mpls ldp discovery

Local LDP Identifier:
    10.15.15.15:0
Discovery Sources:
    Interfaces:
        Ethernet2/2 (ldp): xmit
```

Examples: Preserving QoS Settings with an MPLS LDP Explicit-Null Label

The following example shows how to configure an explicit-null label on an egress LSR, which causes that LSR to advertise the explicit-null label to all adjacent MPLS routers:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# explicit-null
```

The following example shows how to configure an explicit-null label and specify the **for** keyword with a prefix list, which causes all adjacent MPLS router tables to swap an explicit-null label only for those entries specified in the prefix list:

```
switch# configure terminal
switch(config)# ip prefix-list 24 permit 10.24.24.24/32
switch(config)# mpls ldp configuration
switch(config-ldp)# explicit-null for 24
```

The following example shows how to configure an explicit-null label and specify the **to** keyword with a prefix list, which enables you to advertise explicit-null labels only to those adjacent routers specified in the prefix list. To advertise an explicit-null label to a particular router, you must specify the router's LDP ID in the prefix list.

```
switch# configure terminal
switch(config)# ip prefix-list 15 permit 10.15.15.15/32
switch(config)# mpls ldp configuration
switch(config-ldp)# explicit-null to 15
```

The following example shows how to configure an explicit-null label with both the **for** and **to** keywords, which enables you to specify which routes to advertise with explicit-null labels and to which adjacent routers to advertise these explicit-null labels:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# explicit-null for 24 to 15
```

Additional References for MPLS LDP

For additional information related to implementing MPLS LDP, see the following sections:

- Related Documents, page 3-52
- MIBs, page 3-52

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS LDP	MPLS Label Distribution Protocol (LDP)

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS LDP

Table 3-2 lists the release history for this feature.

Table 3-2 Feature History for MPLS LDP

Feature Name	Releases	Feature Information
MPLS LDP	5.2(1)	This feature was introduced.

Configuring MPLS LDP Autoconfiguration

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) autoconfiguration on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LDP Autoconfiguration

The MPLS LDP autoconfiguration feature enables you to globally configure LDP on every interface associated with a specified Interior Gateway Protocol (IGP) instance.

To enable LDP, you should configure it globally and on each interface where it is needed. Configuring LDP on many interfaces can be time consuming.

The MPLS LDP autoconfiguration feature is supported on Open Shortest Path First (OSPF) and Intermediate System-to-System (IS-IS) IGPs. Because you do not have to configure LDP separately on each interface, the autoconfiguration feature makes LDP configuration easier, faster, and error free. If desired, you can also disable LDP on selected interfaces after autoconfiguration is enabled.

Licensing Requirements for MPLS LDP Autoconfiguration

Product	License Requirement
Cisco NX-OS	MPLS LDP autoconfiguration requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LDP Autoconfiguration

MPLS LDP autoconfiguration has the following prerequisites:

- You must enable MPLS LDP.

Guidelines and Limitations for MPLS LDP Autoconfiguration

MPLS LDP autoconfiguration has the following configuration guidelines and limitations:

- This feature is supported only on interfaces that are running OSPF or IS-IS processes. Other IGPs are not supported.
- If you disable LDP globally, autoconfiguration fails and generates a console message explaining that you must first enable LDP globally.
- If MPLS LDP autoconfiguration is configured for an IGP instance, you cannot enter the global **shutdown** command. To disable LDP, you must first enter the **no mpls ldp igp autoconfig command**.
- This feature is not supported on traffic engineering (TE) tunnel interfaces.

Default Settings for MPLS LDP Autoconfiguration

Table 4-1 lists the default settings for MPLS LDP autoconfiguration parameters.

Table 4-1 Default MPLS LDP Autoconfiguration Parameters

Parameters	Default
MPLS LDP autoconfiguration	Disabled

Configuring MPLS LDP Autoconfiguration

This section includes the following topics:

Configuring MPLS LDP Autoconfiguration for OSPF Interfaces

You can configure MPLS LDP autoconfiguration for all interfaces that run OSPF processes. As a result, all interfaces that belong to an OSPF area are enabled for LDP.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that OSPF is enabled. (**You can enable it using the feature ospf command**.)

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. router ospf *process-name*
3. **mpls ldp autoconfig area** *area-id*
4. **(Optional) show mpls ldp discovery detail**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**router ospf** *process-name* **Example:** `switch(config)# router ospf p1` `switch(config-router)#`	Enables an OSPF routing process and enters router configuration mode. You can enter up to 20 alphanumeric characters for the *process-name* argument.
Step 3	**mpls ldp autoconfig area** *area-id* **Example:** `switch(config-router)# mpls ldp autoconfig area 10`	Enables MPLS LDP autoconfiguration for all OSPF interfaces. **For the *area-id* argument, you can specify the area ID as an integer (from 0 to 4,294,967,295) or an IP address.**
Step 4	**show mpls ldp discovery detail** **Example:** `switch(config-router)# show mpls ldp discovery detail`	(Optional) Displays the method used to enable LDP on an interface: • If LDP was enabled on a specific interface, the output displays “Interface config.” • If LDP was enabled using autoconfiguration, the output displays “IGP config.” • If LDP was enabled on a specific interface and using autoconfiguration, the output displays “Interface config, IGP config.”
Step 5	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS LDP Autoconfiguration for IS-IS Interfaces

You can configure MPLS LDP autoconfiguration for all interfaces that run IS-IS processes. As a result, all interfaces that belong to an IS-IS area are enabled for LDP.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that IS-IS is enabled. (**You can enable it using the feature isis command**.)

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. router isis *process-name*
3. **mpls ldp autoconfig {level-1 | level-1-2 | level-2}**

4. **(Optional) show mpls ldp discovery detail**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**router isis** *process-name* **Example:** `switch(config)# router isis p1` `switch(config-router)#`	Enables an IS-IS routing process and enters router configuration mode. You can enter up to 20 alphanumeric characters for the *process-name* argument.
Step 3	**mpls ldp autoconfig {level-1 | level-1-2 | level-2}** **Example:** `switch(config-router)# mpls ldp autoconfig level-1`	Enables MPLS LDP autoconfiguration for all level-1, all level-2, or all level-1 and level-2 IS-IS interfaces.
Step 4	**show mpls ldp discovery detail** **Example:** `switch(config-router)# show mpls ldp discovery detail`	(Optional) Displays the method used to enable LDP on an interface: • If LDP was enabled on a specific interface, the output displays “Interface config.” • If LDP was enabled using autoconfiguration, the output displays “IGP config.” • If LDP was enabled on a specific interface and using autoconfiguration, the output displays “Interface config, IGP config.”
Step 5	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Disabling MPLS LDP Autoconfiguration for Selected OSPF or IS-IS Interfaces

You can disable LDP for specific OSPF or IS-IS interfaces after they were configured with the MPLS LDP autoconfiguration feature.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**

2. **interface ethernet** *slot/port*
3. **no mpls ldp igp autoconfig**
4. **(Optional) show mpls ldp discovery detail**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface ethernet** *slot/port* **Example:** `switch(config)# interface ethernet 7/1` `switch(config-if)#`	Specifies the interface on which you are disabling LDP and enters interface configuration mode.
Step 3	**no mpls ldp igp autoconfig** **Example:** `switch(config-if)# no mpls ldp igp autoconfig`	Disables LDP for the specified interface.
Step 4	**show mpls ldp discovery detail** **Example:** `switch(config-if)# show mpls ldp discovery detail`	(Optional) Displays the method used to enable LDP on an interface. If LDP has been disabled on an interface, that interface does not appear in the output.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS LDP Autoconfiguration

To display the MPLS LDP autoconfiguration, perform one of the following tasks:

Figure 4-1

Command	Purpose
show mpls ldp discovery detail	Displays the method used to enable LDP on an interface.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS LDP Autoconfiguration

This section provides configuration examples for MPLS LDP autoconfiguration and includes the following topics:

Examples: Configuring MPLS LDP Autoconfiguration for OSPF Interfaces

The following example shows how to configure MPLS LDP autoconfiguration for OSPF interfaces and verify the results:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# ip address 10.0.0.13 255.0.0.0
switch(config-if)# exit
switch(config)# router ospf 1
switch(config-router)# area 3 range 10.0.0.13 0.0.255.255
switch(config-router)# mpls ldp autoconfig area 3
switch(config-router)# show mpls ldp discovery detail
Local LDP Identifier:
    10.0.0.13:0
    Discovery Sources:
    Interfaces:
        Ethernet2/1 (ldp): xmit/recv
            Enabled: IGP config;
            Hello interval: 5000 ms; Transport IP addr: 10.0.0.13
            LDP Id: 10.0.0.21:0
              Src IP addr: 168.5.5.21; Transport IP addr: 10.0.0.21
              Hold time: 15 sec; Proposed local/peer: 15/15 sec
              Reachable via 10.0.0.21/32
              Password: not required, none, in use
            Clients: IPv4
        Ethernet2/6 (ldp): xmit/recv
            Enabled: Interface config, IGP config;
            Hello interval: 5000 ms; Transport IP addr: 10.0.0.13
            LDP Id: 10.0.0.22:0
              Src IP addr: 168.6.6.22; Transport IP addr: 10.0.0.22
              Hold time: 15 sec; Proposed local/peer: 15/15 sec
              Reachable via 10.0.0.22/32
              Password: not required, neighbor, in use
            Clients: IPv4
```

The following example shows how to disable LDP on a specific interface after it was enabled using the MPLS LDP autoconfiguration feature:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# no mpls ldp igp autoconfig
switch(config-if)# show mpls ldp discovery detail
Local LDP Identifier:
    10.0.0.13:0
    Discovery Sources:
    Interfaces:
        Ethernet2/6 (ldp): xmit/recv
            Enabled: Interface config, IGP config;
            Hello interval: 5000 ms; Transport IP addr: 10.0.0.13
            LDP Id: 10.0.0.22:0
              Src IP addr: 168.6.6.22; Transport IP addr: 10.0.0.22
```

```
    Hold time: 15 sec; Proposed local/peer: 15/15 sec
    Reachable via 10.0.0.22/32
    Password: not required, neighbor, in use
  Clients: IPv4
```

Examples: Configuring MPLS LDP Autoconfiguration for IS-IS Interfaces

The following example shows how to configure MPLS LDP autoconfiguration for IS-IS interfaces and verify the results:

```
switch# configure terminal
switch(config)# interface ethernet 3/2
switch(config-if)# ip address 10.0.0.30 255.0.0.1
switch(config-if)# ip router isis p1
switch(config-if)# exit
switch(config)# router isis p1
switch(config-router)# mpls ldp autoconfig level-1-2
switch(config-router)# show mpls ldp discovery detail
Local LDP Identifier:
    10.0.0.30:0
    Discovery Sources:
    Interfaces:
        Ethernet3/2 (ldp): xmit/recv
            Enabled: IGP config;
            Hello interval: 5000 ms; Transport IP addr: 10.0.0.30
            LDP Id: 10.0.0.31:0
              Src IP addr: 60.0.0.2; Transport IP addr: 10.0.0.31
              Hold time: 15 sec; Proposed local/peer: 15/15 sec
              Reachable via 10.0.0.31/32
              Password: not required, none, in use
            Clients: IPv4
```

Additional References for MPLS LDP Autoconfiguration

For additional information related to implementing MPLS LDP autoconfiguration, see the following sections:

- Related Documents, page 4-62
- MIBs, page 4-62

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS LDP autoconfiguration	*MPLS LDP Autoconfiguration*

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS LDP Autoconfiguration

Table 4-2 lists the release history for this feature.

Table 4-2 Feature History for MPLS LDP Autoconfiguration

Feature Name	Releases	Feature Information
MPLS LDP autoconfiguration	5.2(1)	This feature was introduced.

Configuring MPLS LDP Session Protection

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) session protection on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LDP Session Protection

The session protection feature provides faster LDP convergence when a link recovers following an outage. It protects an LDP session between directly connected neighbors or an LDP session established for a traffic engineering (TE) tunnel.

MPLS LDP session protection maintains LDP bindings when a link fails. MPLS LDP sessions are protected through the use of LDP hello messages. When you enable MPLS LDP, the label switched routers (LSRs) send messages to find other LSRs with which they can create LDP sessions.

- If the LSR is one hop from its neighbor, it is directly connected to its neighbor. The LSR sends LDP hello messages as User Datagram Protocol (UDP) packets to all the routers on the subnet. The hello message is called an *LDP link hello*. A neighboring LSR responds to the hello message, and the two routers begin to establish an LDP session.
- If the LSR is more than one hop from its neighbor, it is not directly connected to its neighbor. The LSR sends a directed hello message as a UDP packet but as a unicast message specifically addressed to that LSR. The hello message is called an *LDP targeted hello*. The nondirectly connected LSR responds to the hello message, and the two routers establish a targeted LDP session.

MPLS LDP session protection uses LDP targeted hellos to protect LDP sessions. Take, for example, two directly connected routers that have LDP enabled and can reach each other through alternate IP routes in the network. An LDP session that exists between two routers is called an *LDP link session*. When MPLS LDP session protection is enabled, an LDP targeted hello adjacency is also established for the LDP session. If the link between the two routers fails, the LDP link adjacency also fails. However, if the LDP peer is still reachable through IP, the LDP session stays up because the LDP targeted hello adjacency still exists between the routers. When the directly connected link recovers, the session does not need to be reestablished, and LDP bindings for prefixes do not need to be relearned.

Licensing Requirements for MPLS LDP Session Protection

Product	License Requirement
Cisco NX-OS	MPLS LDP session protection requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LDP Session Protection

MPLS LDP session protection has the following prerequisites:

- You must enable MPLS LDP.
- You must enable all routers that participate in MPLS LDP session protection to respond to targeted hellos. Otherwise, the LSRs cannot establish a targeted adjacency. You must configure both neighbor routers for session protection or configure one router for session protection and the other router to respond to targeted hellos.

Default Settings for MPLS LDP Session Protection

Table 5-1 lists the default settings for MPLS LDP session protection parameters.

Table 5-1 Default MPLS LDP Session Protection Parameters

Parameters	Default
MPLS LDP session protection	Disabled

Configuring MPLS LDP Session Protection

You can configure the Cisco NX-OS device for MPLS LDP session protection.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. **session protection** *[***for** *prefix-list] [***duration** *{seconds* | **infinite***}]*
4. **(Optional) show mpls ldp neighbor detail**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 3	**session protection** *[***for** *prefix-list]* *[***duration** *{seconds | **infinite**}]* **Example:** `switch(config-ldp)# session protection for prefix1 duration 100`	Enables MPLS LDP session protection. This command enables LDP sessions to be protected during a link failure. It protects all LDP sessions, unless you specify a prefix list. You can use these keywords to limit the number of protected LDP sessions: • **The for** keyword allows you to specify a prefix list that should be protected. Session protection is then enabled for the peer routers in that prefix list. • **The duration** keyword enables you to specify how long the router should retain the LDP targeted hello adjacency following the loss of the LDP link hello adjacency. The range is from 30 to 2,147,483 seconds. When the link is lost, a timer starts. If the timer expires, the LDP targeted hello adjacency is removed. The **infinite** keyword allows the LDP targeted hello adjacency to exist indefinitely following the loss of the LDP link hello adjacency.
Step 4	**show mpls ldp neighbor detail** **Example:** `switch(config-ldp)# show mpls ldp neighbor detail`	(Optional) Displays the configuration status and current state of MPLS LDP session protection. The state can be “Ready” or “Protecting.” **Note** If the state is “Incomplete,” then the targeted hello adjacency is not yet up.
Step 5	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Clearing an MPLS LDP Session

You can terminate an MPLS LDP session after a link goes down. This procedure is useful when the link needs to be taken out of service or needs to be connected to a different neighbor.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **clear mpls ldp neighbor {* | *neighbor-address*}**

DETAILED STEPS

	Command	Purpose
Step 1	**clear mpls ldp neighbor {* |** *neighbor-address*} **Example:** switch# clear mpls ldp neighbor 10.0.0.13	Clears all LDP neighbor sessions or a specific LDP neighbor session. • **The *** keyword clears all LDP neighbor sessions. • **The** *neighbor-address* **argument** specifies the IP address of the LDP neighbor whose session should be cleared.

Verifying the MPLS LDP Session Protection Configuration

To display the MPLS LDP session protection configuration, perform one of the following tasks:

Command	Purpose
show mpls ldp discovery [detail]	Displays the sources for locally generated LDP targeted hellos.
show mpls ldp neighbor	Displays the status of the LDP session and shows whether the targeted hellos are active.
show mpls ldp neighbor detail	Displays the configuration status and current state of MPLS LDP session protection.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS LDP Session Protection

The following example shows how to configure MPLS LDP session protection for prefix list “prefix1” and verify the results:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# session protection for prefix1 duration 100
switch(config-ldp)# show mpls ldp discovery
 Local LDP Identifier:
    10.0.0.13:0
    Discovery Sources:
    Interfaces:
    Ethernet2/6 (ldp): xmit/recv
            LDP Id: 10.0.0.22:0
    Targeted Hellos:
        10.0.0.13 -> 10.0.0.22 (ldp): active, xmit/recv
```

```
            LDP Id: 10.0.0.22:0
switch(config-ldp)# show mpls ldp neighbor detail
Peer LDP Ident: 10.0.0.22:0; Local LDP Ident 10.0.0.13:0
        TCP connection: 10.0.0.22.36624 - 10.0.0.13.646
        Password: not required, none, in use
        Adj pwd Rx/Tx: [nil]/[nil]
        TCP pwd Rx/Tx: [nil]/[nil]
        State: Oper; Msgs sent/rcvd: 17/20; Downstream; Last TIB rev sent 9
        Up time: 00:10:25; UID: 3; Peer Id 0
        LDP discovery sources:
          Ethernet2/6; Src IP addr: 168.6.6.22
            holdtime: 15000 ms, hello interval: 5000 ms
          Targeted Hello 10.0.0.13 -> 10.0.0.22, active;
            holdtime: infinite, hello interval: 10000 ms
        Addresses bound to peer LDP Ident:
          10.0.0.22       10.0.0.122      2.0.0.73        168.6.6.22
          192.168.1.22
        Peer holdtime: 180000 ms; KA interval: 60000 ms; Peer state: estab
        Clients: Dir Adj Client
        LDP Session Protection enabled, state: Protecting
            duration: 100 seconds
           holdup time remaining: 60 seconds

Capabilities Sent:
          [Dynamic Announcement (0x0506)]
          [Typed Wildcard (0x0970)]
        Capabilities Received:
          [None]
```

Additional References for MPLS LDP Session Protection

For additional information related to implementing MPLS LDP session protection, see the following sections:

- Related Documents, page 5-70
- MIBs, page 5-70

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS LDP session protection	MPLS LDP Session Protection

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS LDP Session Protection

Table 5-2 lists the release history for this feature.

Table 5-2 Feature History for MPLS LDP Session Protection

Feature Name	Releases	Feature Information
MPLS LDP session protection	5.2(1)	This feature was introduced.

Configuring MPLS LDP Lossless MD5 Session Authentication

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) lossless MD5 session authentication on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LDP Lossless MD5 Session Authentication

The MPLS LDP lossless MD5 session authentication feature enables an LDP session to be password protected without tearing down and reestablishing the LDP session.

The following topics provide information about the LDP lossless MD5 session authentication feature:

How Messages Are Exchanged in MPLS LDP Lossless MD5 Session Authentication

MPLS LDP messages (discovery, session, advertisement, and notification messages) are exchanged between LDP peers through two channels:

- LDP discovery messages are transmitted as User Datagram Protocol (UDP) packets to the well-known LDP port.
- Session, advertisement, and notification messages are exchanged through a TCP connection established between two LDP peers. These messages can be protected against spoofed TCP segments by using the TCP MD5 signature option.

The MPLS LDP lossless MD5 session authentication feature allows an LDP session to incur a password change without tearing down and reestablishing the LDP session.

Benefits of MPLS LDP Lossless MD5 Session Authentication

MPLS LDP MD5 session authentication allows you to set up password requirements for a set of LDP neighbors to help prevent unauthorized peers from establishing LDP sessions and to block spoofed TCP messages.

The MPLS LDP lossless MD5 session authentication feature provides these benefits:

- Enables you to specify peers for which password protection is required in order to prevent the establishment of LDP sessions with unexpected peers.
- Enables you to activate or change LDP MD5 session authentication without interrupting the LDP session.
- Enables you to configure multiple passwords so one password can be used now and other passwords later.

Note LDP passwords cannot be configured on interfaces. You can configure one password per peer or per peer group. To configure multiple passwords, you must use keychains. The **key-chain** command allows different key strings to be used at different times according to the keychain configuration.

- Enables you to configure asymmetric passwords, which allows one password to be used for incoming TCP segments and a different password to be used for outgoing TCP segments.
- Enables you to configure passwords so that they overlap for a period of time. This functionality is beneficial when the clocks on two LSRs are not synchronized.

- If the neighboring nodes support graceful restart, then LDP sessions are gracefully restarted. The LDP MD5 password configuration is checkpointed to the standby route processors (RPs). The LDP MD5 password is used by the router when the new active RP attempts to establish LDP sessions with neighbors after the switchover.

Note Passwords can be configured to change over time, but they are not guaranteed to be lossless unless keychains are used with overlapping send and accept lifetimes for the transmit and receive keys.

Keychain Use with MPLS LDP Lossless MD5 Session Authentication

The MPLS LDP lossless MD5 session authentication feature allows keychains to be used to specify different MD5 keys to authenticate LDP traffic exchanged in each direction.

In the following example, three passwords are configured:

- Key 1 specifies the lab password. The **send-lifetime** command enables the lab password to authenticate the outgoing TCP segments from April 2, 2010, at 10:00:00 a.m. until May 2, 2010, at 10:00:00 a.m. The **accept-lifetime** command is configured so that the lab password is never used to authenticate incoming TCP segments. The **accept-lifetime** command enables the lab password for 1 second on January 1, 1970. By setting the date to the past and by enabling a duration of 1 second, the password for incoming TCP segments immediately expires. If the **accept-lifetime** command is omitted from the keychain configuration, then the password is always valid for incoming TCP segments.
- Key 2 and key 3 specify the lab2 and lab3 passwords, respectively. The **send-lifetime** commands enable the passwords for 1 second on January 1, 1970. By setting the date to the past and by enabling a duration of 1 second, the passwords for outgoing TCP segments immediately expire. If the **send-lifetime** commands are omitted from the keychain configuration, the passwords are always valid for outgoing TCP segments. The **accept-lifetime** commands for key 2 and key 3 enable the passwords to authenticate the incoming TCP segments from April 2, 2010, at 10:00:00 a.m. until April 17, 2010, at 10:00:00 a.m. and from April 17, 2010, at 10:00:00 a.m. until May 2, 2010, at 10:00:00 a.m., respectively.

```
switch(config)# ip prefix-list nbrp1 permit 10.0.0.0/32
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string lab
switch(config-keychain-key)# send-lifetime 10:00:00 Apr 2 2010 10:00:00 May 2 2010
switch(config-keychain-key)# accept-lifetime 00:00:00 Jan 1 1970 duration 1
switch(config-keychain-key)# exit
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string lab2
switch(config-keychain-key)# send-lifetime 00:00:00 Jan 1 1970 duration 1
switch(config-keychain-key)# accept-lifetime 10:00:00 Apr 2 2010 10:00:00 Apr 17 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 3
switch(config-keychain-key)# key-string lab3
switch(config-keychain-key)# send-lifetime 00:00:00 Jan 1 1970 duration 1
switch(config-keychain-key)# accept-lifetime 10:00:00 Apr 17 2010 10:00:00 May 2 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password required for nbrp1
switch(config-ldp)# password option 1 for nbrp1 key-chain KeyChain1
```

Application of Rules to Overlapping Passwords

Overlapping passwords can be useful when two LSRs have clocks that are not synchronized. The overlapping passwords provide a window to ensure that TCP packets are not dropped. The following rules apply to overlapping passwords:

- If the send-lifetime value for the next password begins before the send-lifetime value of the current password expires, the password with the shorter key ID is used during the overlap period. The send-lifetime value of the current password can be shortened by configuring a shorter send-lifetime value. Similarly, the send-lifetime value of the current password can be lengthened by configuring a longer send-lifetime value.
- If the accept-lifetime value for the next password begins before the accept-lifetime value of the current password expires, both the next password and the current password are used concurrently. The next password information is passed to TCP. If TCP fails to authenticate the incoming segments with the current password, it tries authenticating with the next password. If TCP authenticates a segment using the new password, it discards the current password and uses the new password from that point on.
- If a password for incoming or outgoing segments expires and no additional valid password is configured, one of the following actions occurs:
  - If a password is required for the neighbor, LDP drops the existing session.
  - If a password is not required for the neighbor, LDP attempts to roll over to a session that does not require authentication. This attempt also fails unless the password expires on both LSRs at the same time.

Resolving LDP Password Problems

LDP displays error messages when an unexpected neighbor attempts to open an LDP session or the LDP password configuration is invalid.

When a password is required for a potential LDP neighbor but no password is configured for it, the LSR ignores LDP hello messages from that neighbor. When the LSR processes the hello message and tries to establish a TCP connection with the neighbor, it displays the error message and stops establishing the LDP session with the neighbor. The error is rate-limited and has the following format:

```
2010 Sep 9 09:59:43.274519 ldp: MD5 protection is required for peer 3.3.3.3:0(default),
but no password is configured.
```

The output of the **show sockets connection detail** command shows a summary of TCP connection failures.

Licensing Requirements for MPLS LDP Lossless MD5 Session Authentication

Product	License Requirement
Cisco NX-OS	MPLS LDP lossless MD5 session authentication requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LDP Lossless MD5 Session Authentication

MPLS LDP lossless MD5 session authentication has the following prerequisites:

- You must configure static or dynamic routing for the LSR.

Guidelines and Limitations for MPLS LDP Lossless MD5 Session Authentication

MPLS LDP lossless MD5 session authentication has the following configuration guidelines and limitations:

- Lossless MD5 session authentication is supported between Cisco NX-OS and Cisco IOS devices.

Default Settings for MPLS LDP Lossless MD5 Session Authentication

Table 6-1 lists the default settings for MPLS LDP lossless MD5 session authentication parameters.

Table 6-1 Default MPLS LDP Lossless MD5 Session Authentication Parameters

Parameters	Default
MPLS LDP lossless MD5 session authentication	Disabled

Configuring MPLS LDP Lossless MD5 Session Authentication

This section includes the following topics:

Configuring MPLS LDP Lossless MD5 Session Authentication Using a Keychain

You configure MPLS LDP lossless MD5 session authentication using a keychain. Keychains allow a different key string to be used at different times according to the keychain configuration. MPLS LDP gives TCP the keychain information, and TCP queries the appropriate keychain to obtain the current live key and key ID for the specified keychain.

If the sessions to be protected are not already using your keychain, the configuration changes take effect the next time that each session is reestablished. For sessions already using this keychain, the configuration changes take effect immediately. For LDP sessions not already using the keychain, the

preexisting authentication remains in effect until the next session is reestablished. A session reestablishment might be forced (with a temporary loss of label switching if LDP graceful restart is not enabled on the session) by using the **clear mpls ldp neighbor** *ip-address* command.

If you are not already using authentication, you must make all the required changes on all peers and then force them into action with the **password required for** *prefix-list* command so that the sessions using the specified the *prefix list* are reestablished using the lossless MD5 session authentication you have defined in your configurations.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **ip prefix-list** *prefix-list* **permit** *network/length*
3. key chain *keychain-name*
4. **key** *key-id*
5. key-string *key*
   (If you plan to configure a fallback keychain in Step 13, repeat Steps 3 through 5 to configure a backup keychain.)
6. **accept-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**}
7. **send-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**}
8. exit
9. exit
10. mpls ldp configuration
11. **(Optional) password required** [**for** p*refix-list*]
12. **password option** *number* **for** *prefix-list* **key-chain** *keychain-name*
13. **(Optional)** password fallback key-chain *keychain-name*
14. **(Optional) show mpls ldp neighbor** [*ip-address* | *interface slot/port*] [**detail**]
15. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**ip prefix-list** *prefix-list* **permit** *network/length* **Example:** `switch(config)# ip prefix-list p1 permit 10.0.0.0/32`	Creates an IP prefix list and specifies the prefixes permitted by the prefix list. The *prefix-list* argument can be up to 63 characters.
Step 3	**key chain** *keychain-name* **Example:** `switch(config)# key chain KeyChain1` `switch(config-keychain)#`	Identifies a group of authentication keys and enters keychain configuration mode.
Step 4	**key** *key-id* **Example:** `switch(config-keychain)# key 1` `switch(config-keychain-key)#`	Identifies an authentication key on a keychain and enters keychain key configuration mode. The *key-id* argument must be a numeral from 0 to 65535.
Step 5	**key-string** *key* **Example:** `switch(config-keychain-key)# key-string pwd1`	Specifies the authentication string for a key. The *string* argument can be from 1 to 80 uppercase or lowercase alphanumeric characters. The first character cannot be a numeral. **Note** If you plan to configure a fallback keychain in Step 13, repeat Steps 3 through 5 to configure a backup keychain.

	Command	Purpose
Step 6	**accept-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**} **Example:** `switch(config-keychain-key)# accept-lifetime 10:00:00 Jan 13 2010 10:00:00 Jun 13 2010`	Specifies the time period during which the authentication key on a keychain can be used for verifying incoming TCP segments. The *start-time* argument identifies the time to start, and the **local** *start-time* argument identifies the time to start in the local time zone. Both arguments have the same parameters: • *hh:mm:ss* is the time format. • Enter the number of days from 1 to 31. • Enter the name of the month. • Enter the year from the present to 2035. **Note** The time reference depends on the clock time zone configuration on the router. If it is configured, the local time zone is used (for example, EST, PST, or so on). Once the start time is entered, select from the following: • The **duration** keyword sets the key lifetime duration in seconds. • The *end-time* argument sets the time to stop. These parameters are the same as those used for the *start-time argument*. • The **infinite** keyword allows the accept-lifetime period to never expire. **Note** If the **no accept-lifetime** value is defined, the associated receive password is valid for authenticating incoming TCP segments.

	Command	Purpose
Step 7	**send-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**} **Example:** `switch(config-keychain-key)# send-lifetime 10:00:00 Jan 13 2010 10:00:00 Jun 13 2010`	Specifies the time period during which the authentication key on a keychain can be used for verifying outgoing TCP segments. The *start-time* argument identifies the time to start, and the **local** *start-time* argument identifies the time to start in the local time zone. Both arguments have the same parameters: • *hh***:***mm***:***ss* is the time format. • Enter the number of days from 1 to 31. • Enter the name of the month. • Enter the year from 1993 to 2035. **Note** The time reference depends on the clock time zone configuration on the router. If it is configured, the local time zone is used (for example, EST, PST, or so on). Once the start time is entered, select from the following: • The **duration** keyword sets the send lifetime duration in seconds. • The *end-time* argument sets the time to stop. These parameters are the same as those used for the *start-time argument*. • The **infinite** keyword allows the send lifetime period to never expire. **Note** If the **no send-lifetime** value is defined, the associated send password is valid for authenticating outgoing TCP segments.
Step 8	`exit` **Example:** `switch(config-keychain-key)# exit` `switch(config-keychain)#`	Exits keychain key configuration mode.
Step 9	`exit` **Example:** `switch(config-keychain)# exit` `switch(config)#`	Exits keychain configuration mode.
Step 10	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.

	Command	Purpose
Step 11	**password required** [**for** *prefix-list*] **Example:** `switch(config-ldp)# password required for p1`	**(Optional)** Specifies that LDP must use a password when establishing a session between LDP peers. The **for** *prefix-list* keyword-argument pair names a prefix list, which specifies that a password is mandatory only for LDP sessions with neighbors whose LDP router IDs are permitted by the list.
Step 12	**password option** *number* **for** *prefix-list* **key-chain** *keychain-name* **Example:** `switch(config-ldp)# password option 25 for p1 key-chain KeyChain1`	Configures an MD5 password for LDP sessions with neighbors whose LDP router IDs are permitted by a specified prefix list. • The *number* argument defines the order in which the prefix lists are evaluated in the determination of a neighbor password. The valid range is from 1 to 32767. • The **for** *prefix-list* keyword-argument pair specifies the name of the prefix list that includes the LDP router IDs of those neighbors for which the password applies. • The **key-chain** *keychain-name* keyword-argument pair specifies a keychain of multiple MD5 keys to be used for the specified LDP sessions.
Step 13	**password fallback key-chain** *keychain-name* **Example:** `switch(config-ldp)# password fallback key-chain KeyChainBackup`	**(Optional)** Configures a backup MD5 keychain for peers that have no keychain configured in Step 12. The **key-chain** *keychain-name* keyword-argument pair specifies a keychain of multiple MD5 keys to be used for the LDP sessions.

	Command	Purpose
Step 14	**show mpls ldp neighbor** [*ip-address* | *interface slot/port*] [**detail**] **Example:** `switch(config-ldp)# show mpls ldp neighbor detail`	**(Optional)** Displays the status of LDP sessions. • The *ip-address* argument identifies the neighbor with the IP address for which password protection is configured. • The *interface* argument lists the LDP neighbors accessible over this interface. • The **detail** keyword displays password information for this neighbor. Here are the items displayed: – An indication as to whether a password is mandatory for this neighbor (required or not required). – The password source (neighbor, fallback, or option number). – An indication as to whether the latest configured password or keychain for this neighbor is used by the TCP session (in use) or the TCP session uses an old password or keychain (stale). A keychain is always considered stale when compared with a simple password, even when the keychain may at the moment lead to using the same simple password.
Step 15	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring a Fallback Password within a Keychain

You can change MD5 passwords for LDP session authentication without having to close and reestablish an existing LDP session by configuring a fallback password within a keychain.

This addition of a fallback password is nondisruptive when done with peers already using the keychain.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. key chain *keychain-name*
3. **key** *key-id*
4. **key-string** *key*

5. **send-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**}
6. **accept-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**}
7. **exit**
8. **key** *key-id*
9. **key-string** *key*
10. **(Optional) show mpls ldp neighbor** [*ip-address* | *interface slot/port*] [**detail**]
11. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**key chain** *keychain-name* **Example:** `switch(config)# key chain KeyChain1` `switch(config-keychain)#`	Identifies a group of authentication keys and enters keychain configuration mode.
Step 3	**key** *key-id* **Example:** `switch(config-keychain)# key 1` `switch(config-keychain-key)#`	Identifies an authentication key on a keychain and enters keychain key configuration mode. The *key-id* argument must be a numeral from 0 to 65535.

	Command	Purpose
Step 4	**key-string** *key* **Example:** `switch(config-keychain-key)# key-string pwd1`	Specifies the authentication string for a key. The *string* argument can be from 1 to 80 uppercase or lowercase alphanumeric characters. The first character cannot be a numeral.
Step 5	**send-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**} **Example:** `switch(config-keychain-key)# send-lifetime 10:00:00 Jan 13 2010 10:00:00 Jun 13 2010`	Specifies the time period during which the authentication key on a keychain can be used for verifying outgoing TCP segments. The *start-time* argument identifies the time to start, and the **local** *start-time* argument identifies the time to start in the local time zone. Both arguments have the same parameters: • *hh*:*mm*:*ss* is the time format. • Enter the number of days from 1 to 31. • Enter the name of the month. • Enter the year from 1993 to 2035. **Note** The time reference depends on the clock time zone configuration on the router. If it is configured, the local time zone is used (for example, EST, PST, or so on). Once the start time is entered, select from the following: • The **duration** keyword sets the send lifetime duration in seconds. • The *end-time* argument sets the time to stop. These parameters are the same as those used for the *start-time argument*. • The **infinite** keyword allows the send lifetime period to never expire. **Note** If the **no send-lifetime** value is defined, the associated send password is valid for authenticating outgoing TCP segments.

	Command	Purpose
Step 6	**accept-lifetime** {*start-time* | **local** *start-time*} {**duration** *seconds* | *end-time* | **infinite**} **Example:** `switch(config-keychain-key)# accept-lifetime 10:00:00 Jan 13 2010 10:00:00 Jun 13 2010`	Specifies the time period during which the authentication key on a keychain can be used for verifying incoming TCP segments. The *start-time* argument identifies the time to start, and the **local** *start-time* argument identifies the time to start in the local time zone. Both arguments have the same parameters: • *hh:mm:ss* is the time format. • Enter the number of days from 1 to 31. • Enter the name of the month. • Enter the year from the present to 2035. **Note** The time reference depends on the clock time zone configuration on the router. If it is configured, the local time zone is used (for example, EST, PST, or so on). Once the start time is entered, select from the following: • The **duration** keyword sets the key lifetime duration in seconds. • The *end-time* argument sets the time to stop. These parameters are the same as those used for the *start-time argument*. • The **infinite** keyword allows the accept-lifetime period to never expire. **Note** If the **no accept-lifetime** value is defined, the associated receive password is valid for authenticating incoming TCP segments.
Step 7	`exit` **Example:** `switch(config-keychain-key)# exit` `switch(config-keychain)#`	Exits keychain key configuration mode.
Step 8	**key** *key-id* **Example:** `switch(config-keychain)# key 65535` `switch(config-keychain-key)#`	Identifies an authentication key on a keychain and enters keychain key configuration mode. The *key-id* argument must be a numeral from 0 to 65535.
Step 9	**key-string** *key* **Example:** `switch(config-keychain-key)# key-string fallback-password`	Specifies the authentication string for a key. The *string* argument can be from 1 to 80 uppercase or lowercase alphanumeric characters. The first character cannot be a numeral.

	Command	Purpose
Step 10	**show mpls ldp neighbor** [*ip-address* | *interface slot/port*] [**detail**] **Example:** `switch(config-ldp)# show mpls ldp neighbor detail`	**(Optional)** Displays the status of LDP sessions. • The *ip-address* argument identifies the neighbor with the IP address for which password protection is configured. • The *interface* argument lists the LDP neighbors accessible over this interface. • The **detail** keyword displays password information for this neighbor. Here are the items displayed: – An indication as to whether a password is mandatory for this neighbor (required or not required) – The password source (neighbor, fallback, or option number) – An indication as to whether the latest configured password for this neighbor is used by the TCP session (in use) or the TCP session uses an old password (stale)
Step 11	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Enabling the Display of MPLS LDP Password Changes

You can enable the display of events related to password configuration changes and rollover events.

Note When a password is required for a neighbor but no password is configured for the neighbor, an error message is logged as shown in the "Resolving LDP Password Problems" section on page 6-75.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. **logging password configuration** [**rate-limit** *number*]
4. **logging password rollover** [**rate-limit** *number*]
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.
Step 3	**logging password configuration [rate-limit** *number***]** **Example:** switch(config-ldp)# logging password configuration rate-limit 20	Enables the display of events related to password configuration changes. The output displays events when a new password is configured or an existing password has been changed or deleted. A rate limit of 1 to 60 messages a minute can be specified.
Step 4	**logging password rollover [rate-limit** *number***]** **Example:** switch(config-ldp)# logging password rollover rate-limit 10	Enables the display of events related to password rollover events. Events are displayed when a new password is used for authentication or when authentication is disabled. A rate limit of 1 to 60 messages a minute can be specified.
Step 5	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS LDP Lossless MD5 Session Authentication

To display the MPLS LDP lossless MD5 session authentication, perform one of these tasks:

Command	Purpose
show mpls ldp neighbor [*ip-address* | *interface slot/port*] **detail**	Displays the status of LDP sessions, including detailed neighbor information.
show mpls ldp neighbor [*ip-address* | *interface slot/port*] **password**	Displays the status of LDP sessions, including password information for the neighbor.

For detailed information about the fields in the output from this command, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS LDP Lossless MD5 Session Authentication

This section provides configuration examples for MPLS LDP lossless MD5 session authentication and includes the following topics:

Examples: Configuring MPLS LDP Lossless MD5 Session Authentication Using a Keychain

The following example shows how to configure two peer LSRs that use symmetrical MD5 keys:

LSR1 (with Router ID 10.1.1.1)

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.2.2.2/32
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string pwd1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:00:00 Feb 1 2010
switch(config-keychain-key)# accept-lifetime 09:00:00 Jan 1 2010 11:00:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password required for 10
switch(config-ldp)# password option 1 for 10 key-chain KeyChain1
```

LSR2 (with Router ID 10.2.2.2)

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.1.1.1/32
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string pwd1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:00:00 Feb 1 2010
switch(config-keychain-key)# accept-lifetime 09:00:00 Jan 1 2010 11:00:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password required for 10
switch(config-ldp)# password option 1 for 10 key-chain KeyChain1
```

The following example shows how to configure two peer LSRs that use asymmetrical MD5 keys:

LSR1 (with Router ID 10.1.1.1)

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.2.2.2/32
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
```

```
switch(config-keychain-key)# key-string pwd1
switch(config-keychain-key)# accept-lifetime 00:00:00 Jan 1 2010 duration 1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:00:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string pwd2
switch(config-keychain-key)# accept-lifetime 09:00:00 Jan 1 2010 11:00:00 Feb 1 2010
switch(config-keychain-key)# send-lifetime 00:00:00 Jan 1 2010 duration 1
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password required for 10
switch(config-ldp)# password option 1 for 10 key-chain KeyChain1
```

LSR2 (with Router ID 10.2.2.2)

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.1.1.1/32
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string pwd2
switch(config-keychain-key)# accept-lifetime 00:00:00 Jan 1 2010 duration 1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:00:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string pwd1
switch(config-keychain-key)# accept-lifetime 09:00:00 Jan 1 2010 11:00:00 Feb 1 2010
switch(config-keychain-key)# send-lifetime 00:00:00 Jan 1 2010 duration 1
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password required for 10
switch(config-ldp)# password option 1 for 10 key-chain KeyChain1
```

Examples: Using a Fallback Password within a Keychain

The following example shows how to configure a fallback password within a keychain. For example, because 65535 is the largest key value allowed for a keychain, the “SampleKeyChain” keychain provides a fallback send and receive password of “fallback-password” for times outside the specified send and accept lifetimes for the specified keystrings (passwords).

```
switch# configure terminal
switch(config)# key chain SampleKeyChain
switch(config-keychain)# key 10
switch(config-keychain-key)# key-string lab1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:30:00 Jan 1 2010
switch(config-keychain-key)# accept-lifetime 10:00:00 Jan 1 2010 10:45:00 Jan 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 20
switch(config-keychain-key)# key-string lab2
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:30:00 Jan 1 2010
switch(config-keychain-key)# accept-lifetime 10:00:00 Jan 1 2010 10:45:00 Jan 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 65535
switch(config-keychain-key)# key-string fallback-password
```

Examples: Common Misconfigurations When Changing an MPLS LDP Lossless MD5 Session Authentication Password

The following examples show common misconfigurations that can occur when the MD5 password is migrated in a lossless way. Misconfigurations can lead to undesired behavior in an LDP session.

- Example: Incorrect Keychain LDP Password Configuration, page 6-91
- Example: Reconfiguring a Keychain to Prevent TCP Authentication and LDP Session Failures, page 6-93
- Avoiding Prefix List Configuration Problems, page 6-94

Example: Incorrect Keychain LDP Password Configuration

Possible misconfigurations can occur when keychain-based commands are used with the **password option for key-chain** command. If the **accept-lifetime** and **send-lifetime** commands are not specified in this configuration, then a misconfiguration can occur when more than two keys are in a keychain.

The following example shows an incorrect keychain configuration with three passwords for LSR A and LSR B in the keychain:

LSR A Incorrect Keychain LDP Password Configuration

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.11.11.11
switch(config)# key chain KeyChain1
switch(config-keychain)# key 10
switch(config-keychain-key)# key-string lab1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:30:00 Jan 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 11
switch(config-keychain-key)# key-string lab2
switch(config-keychain-key)# send-lifetime 10:30:00 Jan 1 2010 10:30:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 12
switch(config-keychain-key)# key-string lab3
switch(config-keychain-key)# send-lifetime 10:30:00 Feb 1 2010 10:30:00 Mar 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password option 5 for 10 key-chain KeyChain1
```

LSR B Incorrect Keychain LDP Password Configuration

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.10.10.10
switch(config)# key chain KeyChain1
switch(config-keychain)# key 10
switch(config-keychain-key)# key-string lab1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:30:00 Jan 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 11
switch(config-keychain-key)# key-string lab2
switch(config-keychain-key)# send-lifetime 10:30:00 Jan 1 2010 10:30:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 12
switch(config-keychain-key)# key-string lab3
switch(config-keychain-key)# send-lifetime 10:30:00 Feb 1 2010 10:30:00 Mar 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
```

```
switch(config)# mpls ldp configuration
switch(config-ldp)# password option 5 for 10 key-chain KeyChain1
```

In the examples above for LSR A and LSR B during the period of the third **send-lifetime 10:30:00 Feb 1 2010 10:30:00 Mar 1 2010 command**, all three configured keys are valid as receive keys, and only the last configured key is valid as a transmit key. The keychain resolution rules dictate that keys 10 and 11 are used as receive keys, and only the last key (key 12) can be used as the transmit key. Because the transmit and receive keys are mismatched, the LDP session will not stay active.

Note When more than two passwords are configured in a keychain, the configuration needs to have both the **accept-lifetime** and **send-lifetime** commands configured correctly.

The following example shows the correct keychain configuration with multiple passwords in the keychain:

LSR A Correct Keychain LDP Password Configuration

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.11.11.11
switch(config)# key chain KeyChain1
switch(config-keychain)# key 10
switch(config-keychain-key)# key-string lab1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:30:00 Jan 1 2010
switch(config-keychain-key)# accept-lifetime 10:00:00 Jan 1 2010 10:45:00 Jan 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 11
switch(config-keychain-key)# key-string lab2
switch(config-keychain-key)# send-lifetime 10:30:00 Jan 1 2010 10:30:00 Feb 1 2010
switch(config-keychain-key)# accept-lifetime 10:15:00 Jan 1 2010 10:45:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 12
switch(config-keychain-key)# key-string lab3
switch(config-keychain-key)# send-lifetime 10:30:00 Feb 1 2010 10:30:00 Mar 1 2010
switch(config-keychain-key)# accept-lifetime 10:15:00 Feb 1 2010 10:45:00 Mar 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
switch(config-ldp)# password option 5 for 10 key-chain KeyChain1
```

LSR B Correct Keychain LDP Password Configuration

```
switch# configure terminal
switch(config)# ip prefix-list 10 permit 10.10.10.10
switch(config)# key chain KeyChain1
switch(config-keychain)# key 10
switch(config-keychain-key)# key-string lab1
switch(config-keychain-key)# send-lifetime 10:00:00 Jan 1 2010 10:30:00 Jan 1 2010
switch(config-keychain-key)# accept-lifetime 10:00:00 Jan 1 2010 10:45:00 Jan 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 11
switch(config-keychain-key)# key-string lab2
switch(config-keychain-key)# send-lifetime 10:30:00 Jan 1 2010 10:30:00 Feb 1 2010
switch(config-keychain-key)# accept-lifetime 10:15:00 Jan 1 2010 10:45:00 Feb 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 12
switch(config-keychain-key)# key-string lab3
switch(config-keychain-key)# send-lifetime 10:30:00 Feb 1 2010 10:30:00 Mar 1 2010
switch(config-keychain-key)# accept-lifetime 10:15:00 Feb 1 2010 10:45:00 Mar 1 2010
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# mpls ldp configuration
```

```
switch(config-ldp)# password option 5 for 10 key-chain KeyChain1
```

In the examples above for LSR A and LSR B during the period of the third **send-lifetime 10:30:00 Feb 1 2010 10:30:00 Mar 1 2010** command, only the last key (key 12) is valid as the transmit and receive key. Therefore, the LDP session remains active.

Example: Reconfiguring a Keychain to Prevent TCP Authentication and LDP Session Failures

If the configuration needs to specify the last key in the keychain to always be valid, then configure the keychain to have at least two keys. Each key must be configured with both the send and accept lifetime period.

```
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string firstpass
switch(config-keychain-key)# accept-lifetime 01:03:00 Sep 10 2010 01:10:00 Sep 10 2010
switch(config-keychain-key)# send-lifetime 01:05:00 Sep 10 2010 01:08:00 Sep 10 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string secondpass
switch(config-keychain-key)# accept-lifetime 01:06:00 Sep 10 2010 01:17:00 Sep 10 2010
switch(config-keychain-key)# send-lifetime 01:08:00 Sep 10 2010 01:15:00 Sep 10 2010
switch(config-keychain-key)# exit
switch(config-keychain)# key 3
switch(config-keychain-key)# key-string thirdpass
```

If the configuration needs to specify the first keychain for the time interval, then use the second key forever after that interval. You can do so by configuring the start time for the second key to begin shortly before the end time of the first key and by configuring the second key to be valid forever after that interval. For example:

```
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string firstpass
switch(config-keychain-key)# accept-lifetime 00:03:00 Sep 10 2010 01:10:00 Sep 10 2010
switch(config-keychain-key)# send-lifetime 00:05:00 Sep 10 2010 01:08:00 Sep 10 2010
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string secondpass
switch(config-keychain-key)# accept-lifetime 01:06:00 Sep 10 2010 infinite
switch(config-keychain-key)# send-lifetime 01:08:00 Sep 10 2010 infinite
```

If the configuration needs to specify the two keys in the order of the second key, first key, and second key again, then specify three keys in that order. For example:

```
switch(config)# key chain KeyChain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string firstpass
switch(config-keychain-key)# accept-lifetime 00:03:00 Sep 10 2010 01:10:00 Sep 10 2010
switch(config-keychain-key)# send-lifetime 00:05:00 Sep 10 2010 01:08:00 Sep 10 2010
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string secondpass
switch(config-keychain-key)# accept-lifetime 01:06:00 Sep 10 2010 01:17:00 Sep 10 2010
switch(config-keychain-key)# send-lifetime 01:08:00 Sep 10 2010 01:15:00 Sep 10 2010
switch(config-keychain)# key 3
switch(config-keychain-key)# key-string firstpass
switch(config-keychain-key)# accept-lifetime 01:13:00 Sep 10 2010 infinite
switch(config-keychain-key)# send-lifetime 01:15:00 Sep 10 2010 infinite
```

Avoiding Prefix List Configuration Problems

Use caution when modifying or deleting a prefix list. Any empty prefix list implies “permit any” by default. When you use the **password option for key-chain** command for MPLS LDP lossless MD5 session authentication, and if the prefix list specified in the command becomes empty as a result of a modification or deletion, then all LDP sessions on the router expect a password. This configuration might cause undesired behavior in LDP sessions. To avoid this scenario, ensure that the proper prefix list is specified for each LSR.

Additional References for MPLS LDP Lossless MD5 Session Authentication

For additional information related to implementing MPLS LDP lossless MD5 session authentication, see the following sections:

- Related Documents, page 6-94
- MIBs, page 6-94

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS LDP lossless MD5 session authentication	MPLS LDP—Lossless MD5 Session Authentication

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS LDP Lossless MD5 Session Authentication

Table 6-2 lists the release history for this feature.

Table 6-2 Feature History for MPLS LDP Lossless MD5 Session Authentication

Feature Name	Releases	Feature Information
MPLS LDP lossless MD5 session authentication	5.2(1)	This feature was introduced.

Configuring MPLS LDP Label Filtering

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) label filtering on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LDP Label Filtering

The following topics provide information about MPLS LDP label filtering:

Local Label Allocation Filtering

This MPLS LDP feature enables you to configure filtering policies for selective local label binding assignments to improve LDP scalability and convergence. This section includes the following topics:

Overview of MPLS LDP Local Label Allocation Filtering

LDP allocates a local label for every route that is learned from the Interior Gateway Protocol (IGP). In the absence of inbound and outbound label filtering, these local labels are advertised to and learned by all LDP peers.

In most Layer 3 virtual private network (VPN) configurations, only the label switched paths (LSPs) created to reach the /32 host routes or Border Gateway Protocol (BGP) next hops between the provider edge (PE) routers carry traffic and are relevant to the Layer 3 VPNs. LSPs between the PE routers that are not members of a VPN use more memory and create additional processing in LDP across the core.

With load increases in the enterprise and service provider domains, scalability has become more important in enterprise and service provider networks. Controlling the local label allocation could offload LDP processing of non-VPN LSPs in the network core routers.

The MPLS LDP local label allocation filtering feature enables you to configure LDP to selectively allocate local labels for a subset of the prefixes learned from the IGP. You can select the LDP allocate local labels for prefixes configured in a prefix list in the global table or for host routes in the global table.

Local label allocation filtering reduces the number of local labels allocated and therefore the number of messages exchanged with peers, which improves LDP scalability and convergence. Figure 7-1 and Figure 7-2 show how controlling local label allocation can reduce the local label space size and greatly reduce the number of advertisements to peers. Figure 7-1 shows the label allocation behavior when LDP allocates a local label for every route and advertises a label binding for every route that is learned from the IGP.

Figure 7-1 Default LDP Local Label Allocation Behavior

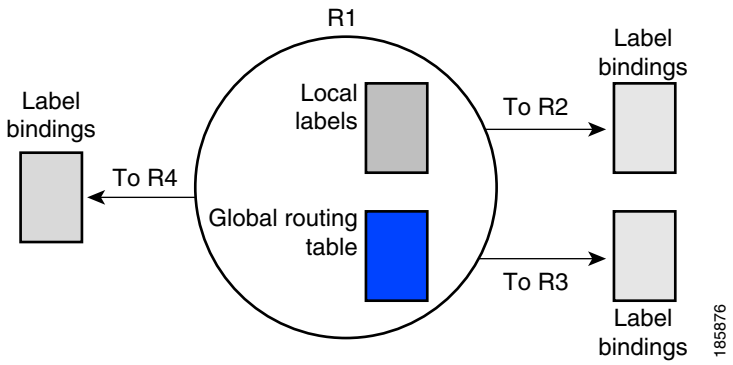


Figure 7-2 shows the LDP behavior with local label allocation control configured. The size of the local label space and the number of label binding advertisements are reduced with local label allocation filtering through the use of a prefix list. The decrease in the number of local labels and label binding

advertisement messages reduces the amount of memory used and improves the convergence time for LDP. The MPLS LDP local label allocation filtering feature also allows for more efficient use of the label space.

Figure 7-2 LDP Behavior with Local Label Allocation Controls

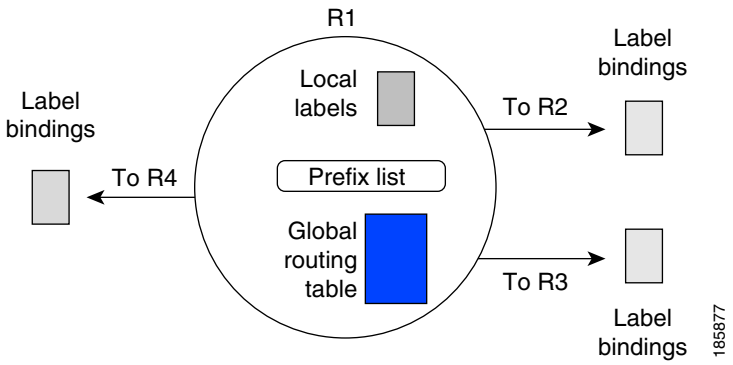


Figure 7-2 shows that router R1 learns the number of routes from its IGP neighbors on routers R2, R3, and R4. A prefix list that is defined on router R1 specifies the prefixes for which LDP allocates a local label.

Note The number of label information base (LIB) entries remains the same regardless of the kind of label filtering used because the remote label bindings for the prefixes that are filtered are kept in the LIB. Memory use is reduced because local label filtering decreases the number of local labels allocated and the number of label bindings advertised to and stored by the peers of an LSR. When all neighboring LSRs are configured with the same local label allocation filter, the number of LIB entries can be reduced (that is, no, or few, LIB entries are created to store remote label bindings).

Prefix Lists for MPLS LDP Local Label Allocation Filtering

The local label allocation filtering feature allows you to configure LDP to allocate local labels for a subset of the learned prefixes. LDP accepts the prefix and allocates a local label if the prefix is permitted by a prefix list. If the prefix list is not defined, LDP accepts all prefixes and allocates local labels based on its default mode of operation.

The benefits of using prefix lists for LDP local label allocation filtering are as follows:

- Prefix lists provide more flexibility for specifying a subset of prefixes and masks.
- Prefix lists use a tree-based matching technique, which is more efficient than evaluating prefixes or host routes sequentially.
- Prefix lists are easy to modify.

Note Prefix lists are also used for outbound label filtering and inbound label filtering. For information on configuring prefix lists, see the "Creating a Prefix List for MPLS LDP Label Filtering" section on page 7-101.

Local Label Allocation Filtering and LDP Actions

Local label allocation filtering modifies the LDP's local label allocation handling. This feature supports local label allocation filtering through the specification of a prefix list or host routes.

With this feature, LDP determines whether a prefix filter is already configured to control the local label allocation on the local node. If a prefix list exists, the local label allocation is confined to the list of prefixes permitted by the configured prefix list.

LDP also responds to local label allocation configuration changes and to configuration changes that affect the prefix list that is used by LDP. Any of the following configuration changes can trigger LDP actions:

- Creating a local label allocation configuration
- Deleting or changing a local label allocation configuration
- Creating a new prefix list for a local label allocation configuration
- Deleting or changing a prefix list for a local label allocation configuration

LDP responds to local label allocation configuration changes by updating the LIB and the forwarding table in the global routing table. To update the LIB after a local label filter configuration change without a session reset, LDP keeps all remote bindings.

If you create a local label allocation configuration without defining a prefix list, no LDP action is required. The local label allocation configuration has no effect because the prefix list is created and permits all prefixes.

If you create or change a prefix list and prefixes that were previously allowed are rejected, LDP goes through a label withdraw and release procedure before the local labels for these prefixes are deallocated.

If you delete a prefix, LDP goes through the label withdraw and release procedure for the LIB local label. If the associated prefix is one for which no LIB entry should be allocated, LDP bypasses this procedure.

Note Local label allocation filtering has no impact on inbound or outbound label filtering because they all provide LDP filtering independently.

Outbound Label Filtering

MPLS LDP supports outbound label binding filtering. You can use this feature to control which label bindings are advertised to LDP neighbors.

Inbound Label Filtering

MPLS LDP supports inbound label binding filtering. You can use this feature to configure prefix lists for controlling the label bindings that an LSR accepts from its peer LSRs. You can limit LDP to accept a set of prefixes from a given LDP neighbor. By default, LDP accepts all labels for all prefixes from all LDP neighbors.

You can use the inbound label binding filtering feature to control the amount of memory used to store LDP label bindings advertised by other routers. For example, in a simple MPLS VPN environment, the VPN PE routers might require an LSP only to their peer PE routers (that is, they do not need LSPs to core routers). Inbound label filtering enables a PE router to accept labels only for other PE routers.

Licensing Requirements for MPLS LDP Label Filtering

Product	License Requirement
Cisco NX-OS	MPLS LDP label filtering requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LDP Label Filtering

MPLS LDP label filtering has the following prerequisites:

- You must enable the MPLS LDP feature.

Guidelines and Limitations for MPLS LDP Label Filtering

MPLS LDP label filtering has the following configuration guidelines and limitations:

- The LDP local label allocation configuration for prefix lists or host routes is supported only in the global routing table.
- A maximum of one local label allocation filter is supported for the global table.
- Wildcard forwarding equivalence class (FEC) requests are not supported with local label allocation filtering.
- Remote label bindings are retained for prefixes even when the allocation of local labels for these prefixes is filtered out (local labels are not allocated).

MPLS LDP outbound label filtering has the following configuration guidelines and limitations:

- To prevent the distribution of any locally assigned labels, use the **no advertise-labels** command with no optional parameters. To reenable the distribution of all locally assigned labels to all LDP neighbors, use the **advertise-labels** command with no optional parameters.
- You can execute multiple **advertise-labels** commands. In the aggregate, such commands determine how the LSR advertises local labels. The following rules describe the effects of multiple commands:
  1. Every **advertise-labels** command has a prefix-pfxlist, peer-pfxlist pair associated with it. The prefix-list pair associated with the **advertise-labels** command (in the absence of both the **for** and **to** keywords) is none, none. The prefix-list pair associated with the **advertise-labels for prefix-pfxlist** command (in the absence of the **to** keyword) is prefix-pfxlist, none.
  2. A given prefix can have, at most, one prefix-pfxlist, peer-pfxlist pair that applies to it, as described as follows:
  a. A given prefix-pfxlist, peer-pfxlist pair applies to a prefix only if the prefix-pfxlist matches the prefix. A match occurs if the prefix-pfxlist permits the prefix.
  b. If more than one prefix-pfxlist, peer-pfxlist pair from multiple **advertise-labels** commands matches a prefix, the prefix-pfxlist, peer-pfxlist pair in the first such command (as determined by the **show running mpls ldp** command) applies to the prefix.
  3. When an LSR is ready to advertise a label for a prefix, the LSR does the following:
  a. Determines whether a prefix-pfxlist, peer-pfxlist pair applies to the prefix.

b. If none applies and the **no advertise-labels** command has been configured, the label for the prefix is not advertised to any peer; otherwise, the label is advertised to all peers.

c. If a prefix-pfxlist, peer-pfxlist pair applies to the prefix and the prefix pfxlist denies the prefix, the label is not advertised to any peer.

d. If the prefix pfxlist permits the prefix and the peer pfxlist is none (that is, the command that applies to the prefix is an **mpls ldp advertise-labels for prefix-list** command without the **to** keyword), the label is advertised to all peers.

e. If the prefix pfxlist permits the prefix and there is a peer pfxlist, the label is advertised to all peers permitted by the peer pfxlist.

Note Typically, LDP advertises labels only for IP prefixes that are in the routing table. You can use the **mpls ldp advertise-labels interface** command to force LDP to advertise a label for a prefix constructed from an interface address and a 32-bit mask. Such a prefix is not usually in the routing table. The output of the **show mpls ip binding detail** command includes the prefix-acl, peer-acl pairs that apply to each prefix.

Default Settings for MPLS LDP Label Filtering

Table 7-1 lists the default settings for MPLS LDP label filtering parameters.

Table 7-1 Default MPLS LDP Label Filtering Parameters

Parameters	Default
MPLS LDP local label allocation filtering	Enabled. Allocate local labels for IGP-learned host routes (/32) only.
MPLS LDP outbound label filtering	Disabled.
MPLS LDP inbound label filtering	Disabled.

Configuring MPLS LDP Label Filtering

This section includes the following topics:

Creating a Prefix List for MPLS LDP Label Filtering

You can create a prefix list for MPLS LDP local label allocation filtering, outbound filtering, or inbound filtering. A prefix list allows LDP to selectively allocate local labels for a subset of the routes learned from the IGP, restrict the advertisement of local labels to specific LDP peers, or control the label bindings that an LSR accepts from its peer LSRs.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **ip prefix-list** prefix-list {**description** description | **seq** number [**deny** network/length [**eq** eq-length | **ge** ge-length | **le** le-length] | **permit** network/length [**eq** eq-length | **ge** ge-length | **le** le-length]] | **deny** network/length [**eq** eq-length | **ge** ge-length | **le** le-length] | **permit** network/length [**eq** eq-length | **ge** ge-length | **le** le-length]}
3. (Optional) **show ip prefix-list [**prefix-list**]**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**ip prefix-list** prefix-list {**description** description | **seq** number [**deny** network/length [**eq** eq-length | **ge** ge-length | **le** le-length] | **permit** network/length [**eq** eq-length | **ge** ge-length | **le** le-length]] | **deny** network/length [**eq** eq-length | **ge** ge-length | **le** le-length] | **permit** network/length [**eq** eq-length | **ge** ge-length | **le** le-length]} **Example:** switch(config)# ip prefix-list p1 permit 10.0.0.2/32 ge 10	Creates a prefix list that you can use as a filter for MPLS LDP label filtering. • The **seq** *number* keyword and argument apply a sequence number to a prefix-list entry. The range for sequence numbers is from 1 to 4,294,967,294. If a sequence number is not entered when this command is configured, a default sequence numbering is applied to the prefix list. The number 5 is applied to the first prefix entry, and subsequent unnumbered entries are incremented by 5. • The **deny** keyword denies access for a matching condition. • The **permit** keyword permits access for a matching condition. • The *network* **and** *length* arguments configure the network address and the length of the network mask in bits, respectively. The network number can be any valid IP address or prefix. The bit mask can be a number from 0 to 32. • The *ge-length* argument specifies the minimum prefix length to be matched. The **ge** keyword represents the greater than or equal to operator. • The *le-length* argument specifies the maximum prefix length to be matched. The **le** keyword represents the less than or equal to operator. • The *eq-length* argument specifies the exact prefix length to be matched. The **eq** keyword represents the equal to operator.
Step 3	**show ip prefix-list [**prefix-list**]** **Example:** switch(config)# show ip prefix-list p1	(Optional) Displays the contents of all current IP prefix lists or of a specified prefix list.
Step 4	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS LDP Local Label Allocation Filtering

You can configure the Cisco NX-OS device for MPLS LDP local label allocation filtering. You can configure a prefix list, host routes, or all routes as a filter for local label allocation.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. **label allocate global {prefix-list** prefix-list | **host-routes | all-routes}**
4. (Optional) **show mpls ldp bindings detail**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 3	**label allocate global** {**prefix-list** *prefix-list* | **host-routes** | **all-routes**} **Example:** `switch(config-ldp)# label allocate global prefix-list p1`	Configures local label allocation filters for MPLS LDP. • The **prefix-list** *prefix-list* keyword and argument specify a prefix list to be used as a filter for MPLS LDP local label allocation. • The **host-routes** keyword specifies that local label allocation be done for host routes only. This is the default configuration. • The **all-routes** keyword specifies that local label allocation be done for all routes.
Step 4	**show mpls ldp bindings detail** **Example:** `switch(config-ldp)# show mpls ldp bindings detail`	(Optional) Displays the filter used for local label allocation. **Note** To see sample output from this command, see the "Sample MPLS LDP Local Label Allocation Filtering Configuration Example" section on page 7-110.
Step 5	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS LDP Outbound Label Filtering

You can configure the Cisco NX-OS device for MPLS LDP outbound label filtering.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **ip prefix-list** prefix-list **permit** network/length
3. **ip prefix-list** prefix-list **permit** network/length

4. mpls ldp configuration
5. **advertise-labels** [**for** prefix-list [**to** prefix-list] | **interface** interface number]
6. (Optional) **show mpls ldp bindings detail**
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**ip prefix-list** *prefix-list* **permit** *network/length* **Example:** switch(config)# ip prefix-list p1 permit 10.0.0.0/32	Creates an IP prefix list and specifies the prefixes permitted by the prefix list. The *prefix-list* argument can be up to 63 characters.
Step 3	**ip prefix-list** *prefix-list* **permit** *network/length* **Example:** switch(config)# ip prefix-list peer1 permit 35.0.0.55/32	Creates an IP prefix list and specifies the prefixes permitted by the prefix list. The *prefix-list* argument can be up to 63 characters.
Step 4	**mpls ldp configuration** **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.
Step 5	**advertise-labels [for** *prefix-list* **[to** *prefix-list***] | interface** *interface number***]** **Example:** switch(config-ldp)# advertise-labels for p1 to peer1	Advertises local labels for some destination prefixes to some LDP peers or advertises local labels for a particular interface. **Note** To block label advertisements to the rest of the LDP peers, use the **no advertise-labels** command.
Step 6	**show mpls ldp bindings detail** **Example:** switch(config-ldp)# show mpls ldp bindings detail	(Optional) Displays the filter used for outbound labels.
Step 7	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS LDP Inbound Label Filtering

You can configure the Cisco NX-OS device for MPLS LDP inbound label filtering.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **ip prefix-list** prefix-list **permit** network/length
3. mpls ldp configuration
4. **neighbor** nbr-address **labels accept** prefix-list
5. (Optional) **show mpls ldp neighbor** [address | interface] [**detail**]
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**ip prefix-list** *prefix-list* **permit** *network/length* **Example:** `switch(config)# ip prefix-list p1 permit 10.0.0.0/32`	Creates an IP prefix list and specifies the prefixes permitted by the prefix list. The *prefix-list* argument can be up to 63 characters.
Step 3	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 4	**neighbor** *nbr-address* **labels accept** *prefix-list* **Example:** `switch(config-ldp)# neighbor 10.12.12.12 labels accept p1`	Specifies the prefix list to be used to filter label bindings for the specified LDP neighbor.
Step 5	**show mpls ldp neighbor [***address* **|** *interface***] [detail]** **Example:** `switch(config-ldp)# show mpls ldp neighbor 10.12.12.12 detail`	(Optional) Displays the filter used for inbound labels.
Step 6	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS LDP Label Filtering Configuration

To display the MPLS LDP label filtering configuration, perform one of the following tasks:

Command	Purpose
show ip prefix-list [*prefix-list]*	Displays the contents of all current IP prefix lists or of a specified prefix list. **Note** It is important that you enter this command to see how the prefix list is defined; otherwise, you cannot verify MPLS LDP label filtering.
show mpls ldp bindings	Displays whether the LSR has remote bindings only from a specified peer for prefixes permitted by the prefix list. **Note** To see sample output from this command, see the "Sample MPLS LDP Local Label Allocation Filtering Configuration Example" section on page 7-110.
show mpls ldp bindings detail	Displays the filter used for local label allocation or for outbound labels.
show mpls ldp neighbor [*address* | *interface*] **[detail]**	Displays the filter used for inbound labels.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS LDP Label Filtering

This section provides configuration examples for MPLS LDP label filtering and includes the following topics:

Examples: Creating a Prefix List for MPLS LDP Local Label Allocation Filtering

The following examples show how to configure a prefix list for MPLS LDP local label allocation filtering.

In the following example, prefix list List1 permits only 192.168.0.0/16 prefixes. LDP accepts 192.168.0.0/16 prefixes but does not assign local labels for the following prefixes: 192.168.0.0/24 and 192.168.2.0/24.

```
switch# configure terminal
switch(config)# ip prefix-list List1 permit 192.168.0.0/16
```

In the following example, prefix list List2 permits a range of prefixes from 192.168.0.0/16 to /20. LDP accepts 192.168.0.0/16 prefixes but does not assign local labels for the following prefixes: 192.168.0.0/24 and 192.168.2.0/24.

```
switch# configure terminal
switch(config)# ip prefix-list List2 permit 192.168.0.0/16 le 20
```

In the following example, prefix list List3 permits a range of prefixes greater than /18. LDP accepts 192.168.17.0/20 and 192.168.2.0/24 prefixes but does not assign a local label for 192.168.0.0/16.

```
switch# configure terminal
switch(config)# ip prefix-list List3 permit 192.168.0.0/16 ge 18
```

Examples: Configuring MPLS LDP Local Label Allocation Filtering

The following examples show how to configure an MPLS LDP local label allocation filter using a prefix list or host routes.

In the following example, a prefix list is configured as the local label allocation filter. Prefix list List3, which permits a range of prefixes greater than /18, is configured as the local label allocation filter for the router. LDP allows 192.168.17.0/20 and 192.168.2.0/24 prefixes but withdraws labels for prefixes not in the allowed range.

```
switch# configure terminal
switch(config)# ip prefix-list List3 permit 192.168.0.0/16 ge 18
switch(config)# mpls ldp configuration
switch(config-ldp)# label allocate global prefix-list List3
```

In the following example, host routes are configured as the local label allocation filter:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# label allocate global host-routes
```

In the following example, all local label allocation filters are removed, and the default LDP local label allocation is restored without a session reset:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# no label allocate global all-routes
```

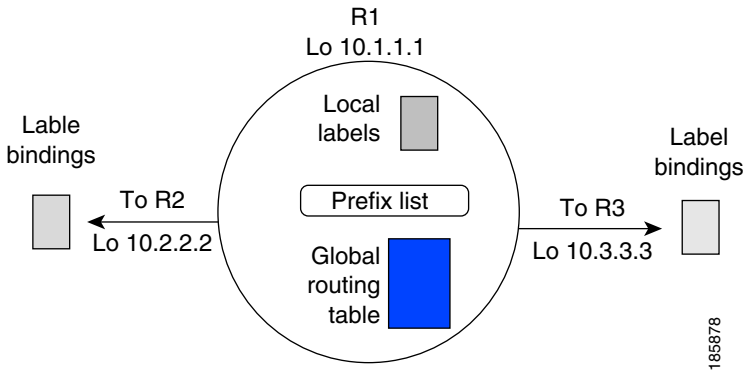
Sample MPLS LDP Local Label Allocation Filtering Configuration Example

Figure 7-3 is a sample configuration used to show how MPLS LDP local label allocation filtering works:

- Routers R1, R2, and R3 have loopback addresses 10.1.1.1, 10.2.2.2, and 10.3.3.3 defined and advertised by the IGP, respectively.
- 10.1.1.1 is the router ID of Router R1, 10.2.2.2 is the router ID of Router R2, and 10.3.3.3 is the router ID of Router R3.
- A prefix list is defined on Router R1 to specify the local labels for which LDP allocates a local label.

Router RI learns a number of routes from its IGP neighbors on Routers R2 and R3.

Figure 7-3 Sample MPLS LDP Local Label Allocation Filtering Configuration Example



You can use LDP commands to verify the following:

- Router R1 has allocated a local label for the correct subset of the prefixes.
- Routers R2 and R3 did not receive any remote bindings for the prefixes for which Router R1 did not assign a local label.

Local Label Bindings on Router R1, Router R2, and Router R3

In the following examples, LDP uses the default behavior of allocating a local label for every route and advertising a label binding for every route learned from the IGP.

The following example shows the contents of the LIB on Router R1 based on the configuration in Figure 7-3:

```
R1# show mpls ldp bindings

  lib entry: 10.1.1.1/32, rev 7
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 16
        remote binding: lsr: 10.2.2.2:0, label: 17
  lib entry: 10.2.2.2/32, rev 13
        local binding:  label: 1000
        remote binding: lsr: 10.3.3.3:0, label: 18
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.3.3.3/32, rev 15
        local binding:  label: 1002
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 18
  lib entry: 10.10.7.0/24, rev 8
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 17
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.10.8.0/24, rev 11
        local binding:  label: 1001
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.10.9.0/24, rev 9
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 16
```

For the first prefix list listed (10.1.1.1/32), Router R1 has received remote labels from Routers R2 and R3 (17 and 16, respectively). The local labels assigned to 10.2.2.2 and 10.3.3.3 on Router R1 (1000 and 1002, respectively) have been advertised to Routers R2 and R3.

The following example shows the contents of the LIB on Router R2 based on the configuration in Figure 7-3:

```
R2# show mpls ldp bindings

  lib entry: 10.1.1.1/32, rev 11
        local binding:  label: 17
        remote binding: lsr: 10.3.3.3:0, label: 16
        remote binding: lsr: 10.1.1.1:0, label: imp-null
  lib entry: 10.2.2.2/32, rev 7
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 18
        remote binding: lsr: 10.1.1.1:0, label: 1000
  lib entry: 10.3.3.3/32, rev 15
        local binding:  label: 18
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: 1002
  lib entry: 10.10.7.0/24, rev 8
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 17
        remote binding: lsr: 10.1.1.1:0, label: imp-null
  lib entry: 10.10.8.0/24, rev 9
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: 1001
  lib entry: 10.10.9.0/24, rev 13
        local binding:  label: 16
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: imp-null
```

For the second prefix list listed (10.2.2.2/32), Router R2 has received remote labels from Routers R1 and R3 (1000 and 18, respectively). The local labels assigned to 10.1.1.1 and 10.3.3.3 on Router R2 (17 and 18, respectively) have been advertised to Routers R1 and R3.

The following example shows the contents of the LIB on Router R3 based on the configuration in Figure 7-3:

```
R3# show mpls ldp bindings

  lib entry: 10.1.1.1/32, rev 13
        local binding:  label: 16
        remote binding: lsr: 10.2.2.2:0, label: 17
        remote binding: lsr: 10.1.1.1:0, label: imp-null
  lib entry: 10.2.2.2/32, rev 15
        local binding:  label: 18
        remote binding: lsr: 10.2.2.2:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: 1000
  lib entry: 10.3.3.3/32, rev 7
        local binding:  label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 18
        remote binding: lsr: 10.1.1.1:0, label: 1002
  lib entry: 10.10.7.0/24, rev 11
        local binding:  label: 17
        remote binding: lsr: 10.2.2.2:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: imp-null
  lib entry: 10.10.8.0/24, rev 8
        local binding:  label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: 1001
  lib entry: 10.10.9.0/24, rev 9
        local binding:  label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 16
        remote binding: lsr: 10.1.1.1:0, label: imp-null
```

For the third prefix list listed (10.3.3.3/32), Router R3 has received remote labels from Routers R1 and R2 (1002 and 18, respectively). The local labels assigned to 10.1.1.1 and 10.2.2.2 on Router R3 (16 and 18, respectively) have been advertised to Routers R1 and R2.

Local Label Allocation Filtering Configuration on Router R1

The following examples show how to configure local label allocation filtering.

The following example shows the selection of host routes as the only filter on Router R1:

```
R1# configure terminal
R1(config)# mpls ldp configuration
R1(config-ldp)# label allocate global host-routes
```

The following example shows how to configure a local label allocation filter that allows or denies prefixes based on prefix list ListA:

```
R1# configure terminal
R1(config)# ip prefix-list ListA permit 0.0.0.0/32 ge 32
R1(config)# mpls ldp configuration
R1(config-ldp)# label allocate global prefix-list ListA
```

Local Label Allocation Filtering Changes Label Bindings on Router R1, Router R2, and Router R3

After configuring a local label allocation filter on Router R1, you can verify the changes in the local label bindings in the LIB on each router. Changes to the output in the LIB entries are highlighted in bold text.

The following example shows how the configuration of a local label allocation prefix-list filter changes the contents of the LIB on Router R1:

```
R1# show mpls ldp bindings

  lib entry: 10.1.1.1/32, rev 7
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 16
        remote binding: lsr: 10.2.2.2:0, label: 17
  lib entry: 10.2.2.2/32, rev 13
        local binding:  label: 1000
        remote binding: lsr: 10.3.3.3:0, label: 18
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.3.3.3/32, rev 15
        local binding:  label: 1002
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 18
  lib entry: 10.10.7.0/24, rev 8
        no local binding
        remote binding: lsr: 10.3.3.3:0, label: 17
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.10.8.0/24, rev 11
        no local binding
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.10.9.0/24, rev 9
        no local binding
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 16
```

The following example shows how the configuration of a local label allocation prefix-list filter on Router R1 changes the contents of the LIB on Router R2:

```
R2# show mpls ldp bindings

  lib entry: 10.1.1.1/32, rev 11
        local binding:  label: 17
        remote binding: lsr: 10.3.3.3:0, label: 16
  lib entry: 10.2.2.2/32, rev 7
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 18
        remote binding: lsr: 10.1.1.1:0, label: 1000
  lib entry: 10.3.3.3/32, rev 15
        local binding:  label: 18
        remote binding: lsr: 10.3.3.3:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: 1002
lib entry: 10.10.7.0/24, rev 8
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: 17
  lib entry: 10.10.8.0/24, rev 9
        local binding:  label: imp-null
        remote binding: lsr: 10.3.3.3:0, label: imp-null
  lib entry: 10.10.9.0/24, rev 13
        local binding:  label: 16
        remote binding: lsr: 10.3.3.3:0, label: imp-null
```

The 10.10.7.0/24, 10.10.8.0/24, and 10.10.9.0/24 prefixes are no longer assigned with local labels. Therefore, Router R1 sends no label advertisement for these prefixes.

The following example shows how the configuration of a local label allocation prefix-list filter on Router R1 changes the contents of the LIB on Router R3:

```
R3# show mpls ldp bindings

  lib entry: 10.1.1.1/32, rev 13
        local binding:  label: 16
        remote binding: lsr: 10.2.2.2:0, label: 17
        remote binding: lsr: 10.1.1.1:0, label: imp-null
  lib entry: 10.2.2.2/32, rev 15
        local binding:  label: 18
        remote binding: lsr: 10.2.2.2:0, label: imp-null
        remote binding: lsr: 10.1.1.1:0, label: 1000
  lib entry: 10.3.3.3/32, rev 7
        local binding:  label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 18
        remote binding: lsr: 10.1.1.1:0, label: 1002
  lib entry: 10.10.7.0/24, rev 11
        local binding:  label: 17
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.10.8.0/24, rev 8
        local binding:  label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: imp-null
  lib entry: 10.10.9.0/24, rev 9
        local binding:  label: imp-null
        remote binding: lsr: 10.2.2.2:0, label: 16
```

The 10.10.7.0/24, 10.10.8.0/24, and 10.10.9.0/24 prefixes are no longer assigned with local labels. Again, Router R1 sends no label advertisement for these prefixes.

Examples: Configuring MPLS LDP Outbound Label Filtering

The following example shows how to configure the device to advertise the label for network 10.0.0.0 only to LSR 35.0.0.55 and the labels for all other prefixes to all LSRs:

```
switch# configure terminal
switch(config)# ip prefix-list pfx1 permit 10.0.0.0/32
switch(config)# ip prefix-list peer1 permit 35.0.0.55/32
switch(config)# mpls ldp configuration
switch(config)# no advertise-labels
switch(config-ldp)# advertise-labels for pfx1 to peer1
```

Examples: Configuring MPLS LDP Inbound Label Filtering

The following example shows how to configure a prefix list to filter label bindings received on sessions with the neighbor 10.0.0.31.

Label bindings for prefixes that match 10.b.c.d are accepted, where b is less than or equal to 63, and c and d can be any integer between 0 and 128. Other label bindings received from 10.0.0.31 are rejected.

```
switch# configure terminal
switch(config)# ip prefix-list 1 permit 10.0.0.0/10 le 32
switch(config)# mpls ldp configuration
switch(config-ldp)# neighbor 10.0.0.31 labels accept 1
switch(config-ldp)# show mpls ldp neighbor 10.0.0.31 detail

Peer LDP Ident: 10.0.0.31:0; Local LDP Ident 10.0.0.30:0
        TCP connection: 10.0.0.31.18303 - 10.0.0.30.646
        Password: not required, none, in use
        Adj pwd Rx/Tx: [nil]/[nil]
        TCP pwd Rx/Tx: [nil]/[nil]
        State: Oper; Msgs sent/rcvd: 28987/28988; Downstream; Last TIB rev sent 57
        Up time: 2w3d; UID: 16; Peer Id 0
        LDP discovery sources:
          Ethernet2/2; Src IP addr: 60.0.0.2
            holdtime: 15000 ms, hello interval: 5000 ms
        Addresses bound to peer LDP Ident:
          10.0.0.31       3.3.151.13      60.0.0.2        61.0.0.1
        Peer holdtime: 180000 ms; KA interval: 60000 ms; Peer state: estab
        Capabilities Sent:
          [Dynamic Announcement (0x0506)]
          [Typed Wildcard (0x0970)]
        Capabilities Received:
          [Dynamic Announcement (0x0506)]
          [Typed Wildcard (0x0970)]
```

The following example shows label bindings that were learned from 10.0.0.31. It verifies that the LIB does not contain label bindings for prefixes that have been excluded.

```
switch# show mpls ldp bindings neighbor 10.0.0.31

lib entry: 3.3.0.0/16, rev 57
        remote binding: lsr: 10.0.0.31:0, label: imp-null
lib entry: 10.0.0.30/32, rev 2
        remote binding: lsr: 10.0.0.31:0, label: 17
lib entry: 10.0.0.31/32, rev 39
        remote binding: lsr: 10.0.0.31:0, label: imp-null
lib entry: 10.0.0.32/32, rev 37
        remote binding: lsr: 10.0.0.31:0, label: 16
```

```
lib entry: 60.0.0.0/8, rev 55
        remote binding: lsr: 10.0.0.31:0, label: imp-null
lib entry: 61.0.0.0/8, rev 56
```
remote binding: lsr: 10.0.0.31:0, label: imp-null

Additional References for MPLS LDP Label Filtering

For additional information related to implementing MPLS LDP label filtering, see the following sections:

- Related Documents, page 7-116
- MIBs, page 7-116

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS LDP label filtering	MPLS LDP—Local Label Allocation Filtering

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS LDP Label Filtering

Table 7-2 lists the release history for this feature.

Table 7-2 Feature History for MPLS LDP Label Filtering

Feature Name	Releases	Feature Information
MPLS LDP local label allocation filtering	5.2(1)	This feature was introduced.
MPLS LDP outbound label filtering	5.2(1)	This feature was introduced.
MPLS LDP inbound label filtering	5.2(1)	This feature was introduced.

Configuring MPLS Static Label Binding

This chapter describes how to configure Multiprotocol Label Switching (MPLS) static label binding on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS Static Label Binding

The MPLS static label binding feature enables you to configure:

- MPLS static labels to be used in the core of the MPLS network instead of dynamic labels. Because these labels are exchanged by LDP, they are not VRF-aware as LDP runs only the default VRF on Cisco NX-OS.

- MPLS virtual private network (VPN) static labels to be used at the edge of an MPLS VPN. These labels are VRF-aware because they correspond to a prefix in a particular VRF.

Note Cisco NX-OS does not support the allocation of aggregate MPLS VPN static labels on a per-VRF basis.

The following topics provide information about MPLS static labels:

- Overview of MPLS Static Labels and LDP VRF-Aware Static Labels, page 8-119
- Labels Reserved for Static Assignment, page 8-119

Overview of MPLS Static Labels and LDP VRF-Aware Static Labels

Label switch routers (LSRs) dynamically learn the labels that they should use to label-switch packets by using the following label distribution protocols:

- Label Distribution Protocol (LDP), which is the Internet Engineering Task Force (IETF) standard used to bind labels to network addresses
- Resource Reservation Protocol (RSVP), which is used to distribute labels for traffic engineering (TE)
- Border Gateway Protocol (BGP), which is used to distribute labels for MPLS VPNs

The LSR installs the dynamically learned label into its label forwarding information base (LFIB).

You can configure static labels for the following purposes:

- To bind labels to IPv4 prefixes to support MPLS hop-by-hop forwarding through neighbor routers that do not implement LDP label distribution. MPLS static labels allow you to configure entries in the MPLS forwarding table and assign label values to forwarding equivalence classes (FECs) that are learned by IGP. You can manually configure an LSP without setting up an LDP session between the endpoints.
- To statically bind a VRF-aware label on a provider edge (PE) router to a customer network prefix (VPN IPv4 prefix). You can use VRF-aware static labels with non-default VRF tables so that you can use the labels at the VPN edge. VRF-aware static labels are advertised by VPNv4 BGP in the backbone.

Labels Reserved for Static Assignment

Before you can manually assign labels, you must reserve a range of labels to be used for the manual assignment. Reserving the labels ensures that the labels are not dynamically assigned.

Licensing Requirements for MPLS Static Label Binding

Product	License Requirement
Cisco NX-OS	MPLS static label binding requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS Static Label Binding

MPLS static label binding has the following prerequisites:

- You must enable MPLS LDP.

Guidelines and Limitations for MPLS Static Label Binding

MPLS static label binding has the following configuration guidelines and limitations:

- Static local labels preempt dynamically allocated labels.
- Configuring static local labels for connected routes is not supported.
- LDP-learned remote labels are preferred over locally configured outgoing labels.
- Static outgoing labels should match the next-hop's incoming labels.
- VRF is not supported for static outgoing labels.
- LDP is not supported in non-default VRFs.

Default Settings for MPLS Static Label Binding

Table 8-1 lists the default settings for MPLS static label binding parameters.

Table 8-1 Default MPLS Static Label Binding Parameters

Parameters	Default
MPLS static label binding	Disabled

Configuring MPLS Static Label Binding

This section includes the following topics:

Reserving Labels to Use for MPLS Static Labels and LDP VRF-Aware Static Labels

You can reserve the labels that are to be statically assigned so that they are not dynamically assigned.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled.

SUMMARY STEPS

1. **configure terminal**
2. mpls label range *min-value max-value* [**static** *min-static-value max-static-value*]
3. (Optional) **show mpls label range**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls label range** *min-value max-value* [**static** *min-static-value max-static-value*] **Example:** `switch(config)# mpls label range 200 100000 static 16 199`	Reserves a range of labels for static label assignment. For Cisco NX-OS releases prior to 6.1, the range for the minimum and maximum values is from 16 to 492286. Beginning with Cisco NX-OS Release 6.1, the range is from 16 to 471804.
Step 3	**show mpls label range** **Example:** `switch(config)# show mpls label range`	(Optional) Displays information about the range of values for local labels, including those labels that are available for static assignments.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS Static Labels in the MPLS VPN Provider Core

You can configure MPLS static labels in order to configure entries in the MPLS forwarding table and assign label values to FECs learned by routing protocols or static routes. You can manually configure an LSP without setting up LDP sessions between the endpoints.

Note In MPLS VPN networks, you can use static labels only in the MPLS VPN provider core.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on each LSR.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration

3. **mpls static binding ipv4** *prefix mask {label* | **input** *label* | **output** *nexthop {***explicit-null** | **implicit-null** | *label}}*
4. (Optional) **show mpls static binding ipv4**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** switch(config)# mpls ldp configuration switch(config-ldp)#	Enters LDP configuration mode.
Step 3	**mpls static binding ipv4** *prefix mask* **{***label* | **input** *label* | **output** *nexthop* **{explicit-null | implicit-null |** *label***}}** **Example:** switch(config-ldp)# mpls static binding ipv4 10.2.2.0 255.255.255.255 input 17	**Specifies the static binding of labels to IPv4 prefixes.** **Specified bindings are installed automatically in the MPLS forwarding table as routing demands.**
Step 4	**show mpls static binding ipv4** **Example:** switch(config-ldp)# show mpls static binding ipv4	(Optional) Displays the configured static labels.
Step 5	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS LDP VRF-Aware Static Labels at the Edge of the VPN

You can statically bind a VRF-aware label on a PE router to a customer network prefix (VPN IPv4 prefix). You can use VRF-aware static labels with non-default VRF tables so the labels can be used at the VPN edge.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on each LSR.

Ensure that MPLS VPN is configured.

Ensure that the provider network has MPLS LDP installed and running.

SUMMARY STEPS

1. **configure terminal**

2. mpls ldp configuration
3. **mpls static binding ipv4 vrf** *vrf_name prefix mask {***input** *label | label}*
4. (Optional) **show mpls static binding ipv4 vrf** *vrf_name*
5. (Optional) **show running-config vrf** *vrf_name*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.
Step 3	**mpls static binding ipv4 vrf** *vrf_name prefix mask* **{input** *label* **|** *label***}** **Example:** `switch(config-ldp)# mpls static binding ipv4 vrf vrf1 10.2.0.0 255.255.0.0 input 17`	Binds a prefix to a local label. Specified bindings are installed automatically in the MPLS forwarding table as routing demands. **Note** You must configure the MPLS VPN and VRFs before creating VRF-aware static labels.
Step 4	**show mpls static binding ipv4 vrf** *vrf_name* **Example:** `switch(config-ldp)# show mpls static binding ipv4 vrf vrf1`	(Optional) Displays the configured MPLS static bindings.
Step 5	**show running-config vrf** *vrf_name* **Example:** `switch(config-ldp)# show running-config vrf vrf1`	(Optional) Displays the VRF running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-ldp)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS Static Label Binding Configuration

To display the MPLS static label binding configuration, perform one of the following tasks:

Command	Purpose
show mpls label range	Displays information about the range of values for local labels, including those labels that are available for static assignments.
show mpls ldp bindings	Displays the MPLS LDP label information base (LIB).
show mpls static binding	Displays the configured static label bindings.
show mpls switching	Displays the MPLS label switching database.
show running-config vrf *vrf_name*	Displays the VRF running configuration.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS Static Label Binding

This section provides configuration examples for MPLS static label binding and includes the following topics:

Examples: Reserving Labels to Use for MPLS Static Labels and MPLS LDP VRF-Aware Static Labels

The following example shows how to reserve a generic range of labels from 200 to 100000 and configure a static label range of 16 to 199:

```
switch(config)# mpls label range 200 100000 static 16 199
```

In this example, the output from the **show mpls label range** command displays the new label range:

```
switch# show mpls label range
Downstream Generic label region: Min/Max label: 200/100000
Range for static labels: Min/Max/Number: 16/199
```

Examples: Configuring MPLS Static Labels in the MPLS VPN Provider Core

The following example shows how to configure input and output labels for several prefixes:

```
switch(config)# mpls ldp configuration
switch(config-ldp)# mpls static binding ipv4 10.0.0.0 255.0.0.0 55
switch(config-ldp)# mpls static binding ipv4 10.0.0.0 255.0.0.0 output 10.0.0.66 167
switch(config-ldp)# mpls static binding ipv4 10.66.0.0 255.255.0.0 input 17
switch(config-ldp)# mpls static binding ipv4 10.66.0.0 255.255.0.0 output 10.13.0.8
explicit-null
```

The **show mpls static binding ipv4** command displays the configured static labels:

```
switch# show mpls static binding ipv4
10.0.0.0/8: (vrf: default) Incoming label: 55
  Outgoing labels:
    10.0.0.66   167
10.66.0.0/16: (vrf: default) Incoming label: 17
  Outgoing labels:
    10.13.0.8.0
```

Examples: Configuring MPLS LDP VRF-Aware Static Labels at the VPN Edge

The following example shows how to configure static label bindings and input (local) labels for various prefixes:

```
switch(config)# mpls ldp configuration
switch(config-ldp)# mpls static binding ipv4 vrf vpn100 10.0.0.0 255.0.0.0 55
switch(config-ldp)# mpls static binding ipv4 vrf vpn100 10.66.0.0 255.255.0.0 input 17
```

In the following output, the **show mpls static binding ipv4 vrf** command displays the configured VRF-aware static bindings:

```
switch# show mpls static binding ipv4 vrf vpn100
10.0.0.0/8:   (vrf: vpn100)  Incoming label: 55
  Outgoing labels:  None
10.66.0.0/16:   (vrf: vpn100)  Incoming label: 17
  Outgoing labels:  None
```

Additional References for MPLS Static Label Binding

For additional information related to implementing MPLS static label binding, see the following sections:

- Related Documents, page 8-126
- MIBs, page 8-126

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS static label binding	MPLS LDP—VRF-Aware Static Labels

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS Static Label Binding

Table 8-2 lists the release history for this feature.

Table 8-2 Feature History for MPLS Static Label Binding

Feature Name	Releases	Feature Information
MPLS static label binding	6.1(1)	Changed the maximum value for the MPLS static label range to 471804.
MPLS static label binding	5.2(1)	This feature was introduced.

Configuring MPLS LDP Graceful Restart

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) graceful restart on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LDP Graceful Restart

MPLS LDP graceful restart is an LDP protocol feature that, in the presence of a temporary LDP control plane disruption, preserves Nonstop Forwarding (NSF) support of data plane traffic being carried on label-switched paths (LSPs).

An example of an LDP control plane disruption is when the LDP control plane on a router restarts. With MPLS LDP graceful restart, that router and all of its neighbors preserve their forwarding state so that traffic continues to be forwarded along the LDP LSPs. As the LDP control plane restarts, that router and its neighbors use graceful restart procedures to transition back to normal control plane operation. The result is that the disruption associated with a control plane restart is greatly reduced.

This section includes the following topics:

Introduction to MPLS LDP Graceful Restart

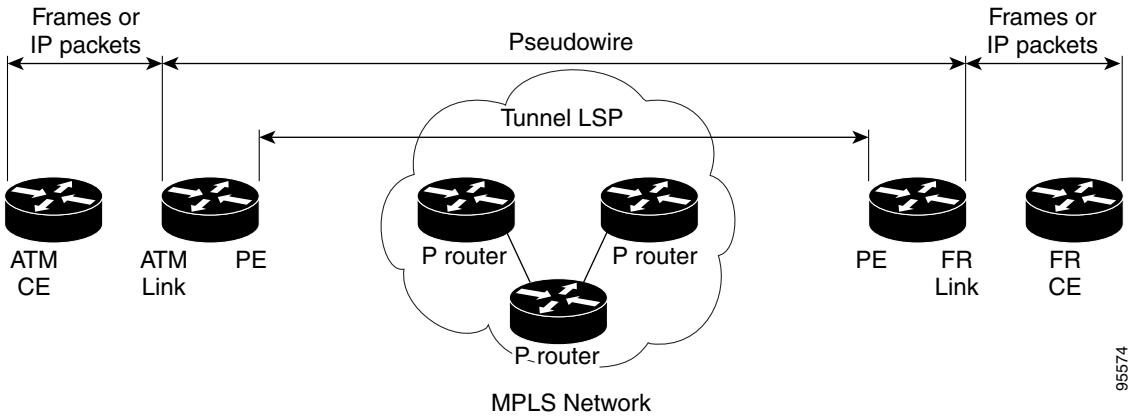
MPLS LDP graceful restart preserves data plane forwarding along LDP LSPs in the presence of temporary LDP control plane disruption. The disruption might be an LDP control plane restart caused by a supervisor switchover or process restart or a TCP or UDP event that disrupts control plane communication between two LDP control planes, even if neither restarts.

The following example describes how MPLS LDP graceful restart operates in the presence of an LDP control plane restart. Note that this interaction has two different roles: the router with the restarting LDP control plane and the neighbor router that detects a loss and recovery of its LDP session to the restarting router. Each carries out MPLS LDP graceful restart procedures appropriate to its role.

The topology shown in Figure 9-1 has the following elements:

- All three routers are running MPLS LDP, all with graceful restart enabled.
- An LDP session exists between R2 and R1, and another LDP session exists between R2 and R3.
- LDP LSPs have been established, including LSPs that connect R1 and R3 and are carrying data traffic. In a network with more routers, multiple LSPs might traverse R1-R2-R3 in both directions.

Figure 9-1 Example of a Network Using LDP Graceful Restart



The following sequence shows how the three routers cooperate to provide NSF and avoid a disruption to data traffic:

1. At session establishment, each router reports to its neighbors that it supports graceful restart. Each session endpoint knows that both ends of the session support graceful restart.

2. R2 begins a supervisor switchover. R2's LDP control plane and all of its LDP TCP connections and LDP UDP hello adjacencies stop operating. R2's data plane marks all of its label entries as stale but continues to use those entries for forwarding MPLS data traffic.
3. R1 notices a loss of communication with R2. (In this sequence, R3 performs the same actions as R1.) R1 marks all of its label bindings from R2 as stale but continues to use those entries for forwarding data traffic.
4. R1 and R2 reestablish an LDP session. On R1, entering the **show mpls ldp neighbor graceful-restart** command displays information about the recovering session.
5. R2 reacquires its local label binding information. R2's data plane typically provides R2's control plane with the same local label for each prefix that was used prior to the restart.
6. Both routers readvertise their label binding information. If R1 relearns a label from R2, R1 marks the binding as no longer stale. If R2 learns a label from R1 and submits that label to its data plane, the data plane marks the entry as no longer stale.
7. After a certain amount of time has passed, R1's LDP control plane cleans up all entries that are still marked as stale. Similarly, R2's data plane removes all entries that are still marked as stale.

Typically, if no other network disruption occurs during this graceful restart operation, all bindings are relearned from the neighbor with the same label values as before the session restart. In this scenario, all saved bindings are marked as not stale during recovery, and no entries need to be cleaned up.

Another scenario of interest is a TCP or UDP communication failure without an LDP control plane restart. In this case, the two LDP control planes at either end of the session detect the communication failure. Each LDP control plane applies the same procedures as R1 used above, and NSF is achieved. R1 carries out the same MPLS LDP graceful restart procedures whether the communication failure with R2 was caused by a restart of R2's control plane or by a networking issue without a restart of R2's control plane.

You can set various timers to limit how long the routers wait for an LDP session to be reestablished before restarting the router.

What Happens if a Router Does Not Have MPLS LDP Graceful Restart Enabled

When a router that does not support MPLS LDP graceful restart undergoes a control plane restart, its data plane MPLS forwarding entries are freed.

A neighbor of such a router, detecting a loss of communication, frees all bindings from the restarting router. This behavior occurs whether the neighbor supports MPLS LDP graceful restart or not. A neighbor that supports MPLS LDP graceful restart learns at session establishment that the restarting router is not supporting MPLS LDP graceful restart. The neighbor does not run graceful restart procedures when detecting a loss of communication to the restarting router.

The cleanup actions of both the restarting router and its neighbor cause any data traffic on the old LSPs to be dropped until recovery activities have addressed the situation. Recovery activities include traffic reroute or the reestablishment of MPLS LDP LSPs or both.

How a Router Advertises that it Supports MPLS LDP Graceful Restart

A router that supports MPLS LDP graceful restart announces its capabilities to its neighbors in the fault tolerant (FT) type-length-value (TLV) in the LDP initialization message. The router sends the LDP initialization message to a neighbor as part of establishing an LDP session.

The FT session TLV includes the following information:

- The Learn from Network (L) flag is set to 1, which indicates that the router is configured to perform MPLS LDP graceful restart.
- The Reconnect Timeout field shows the time (in milliseconds) that the neighbor should wait for a reconnection if the LDP session is lost. If the timer is set to 0 and the local router fails, its peers do not wait for it to recover.
- The Recovery Time field shows the time (in milliseconds) that the neighbor should retain the MPLS forwarding state during a recovery currently in progress. For example, if a neighbor restarted and did not preserve the MPLS forwarding state across the restart, the neighbor should set its recovery time to 0.

Note The reconnect time applies to the next communication loss while the recovery time applies to the recovery from the preceding communication loss.

Licensing Requirements for MPLS LDP Graceful Restart

Product	License Requirement
Cisco NX-OS	MPLS LDP graceful restart requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LDP Graceful Restart

MPLS LDP graceful restart has the following prerequisites:

- You must enable MPLS LDP on the device.
- Ensure that MPLS LDP graceful restart has not been disabled on the device.
- For Nonstop Forwarding (NSF) to occur, MPLS LDP graceful restart must be present on the restarting router and its peers.

Default Settings for MPLS LDP Graceful Restart

Table 9-1 lists the default settings for MPLS LDP graceful restart parameters.

Table 9-1 Default MPLS LDP Graceful Restart Parameters

Parameters	Default
MPLS LDP graceful restart	Enabled
Forwarding state holding time	600 seconds
Max recovery time	120 seconds
Neighbor liveness timer	120 seconds

Configuring MPLS LDP Graceful Restart

The MPLS LDP graceful restart feature is globally disabled or enabled.

You must not disable MPLS LDP graceful restart on the routers in order for forwarding to be preserved during an interruption in service.

When you disable or enable LDP graceful restart, it has no effect on existing LDP sessions. This configuration change applies to new sessions that are established after the change.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Ensure that MPLS LDP is enabled on the device.

SUMMARY STEPS

1. **configure terminal**
2. mpls ldp configuration
3. [**no**] **graceful-restart** [**timers** {**forwarding-holding** *seconds* | **max-recovery** *seconds* | **neighbor-liveness** *seconds}*]
4. (Optional) **show mpls ldp graceful-restart**
5. (Optional) show mpls ldp neighbor graceful restart
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters LDP configuration mode.

	Command	Purpose
Step 3	**[no] graceful-restart** [**timers** {**forwarding-holding** *seconds* | **max-recovery** *seconds* | **neighbor-liveness** *seconds*}] **Example:** **switch(config-ldp)# graceful-restart timers max-recovery 100**	Disables or enables the router to protect the LDP bindings and MPLS forwarding state during a disruption in service. MPLS LDP graceful restart is enabled by default. (Optional) You can configure the following graceful-restart timers if the default values need to be changed: • **forwarding-holding**—Specifies the amount of time that the MPLS forwarding state should be preserved after the control plane restarts. The range is from 30 to 600 seconds. • **max-recovery**—Specifies the amount of time that a router should hold stale label-FEC bindings after an LDP session has been reestablished. The range is from 15 to 600 seconds. • **neighbor-liveness**—Specifies the amount of time that a router should wait for an LDP session to be reestablished. The range is from 5 to 300 seconds.
Step 4	**show mpls ldp graceful-restart** **Example:** **switch(config-ldp)# show mpls ldp** graceful-restart	(Optional) Displays this router's LDP graceful-restart configuration.
Step 5	show mpls ldp neighbor graceful-restart **Example:** **switch(config-ldp)# show mpls ldp neighbor** graceful-restart	(Optional) Displays the graceful-restart parameters for this router's sessions with its LDP neighbors.
Step 6	**copy running-config startup-config** **Example:** switch(config-ldp)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS LDP Graceful Restart Configuration

To display the MPLS LDP graceful restart configuration, perform one of the following tasks:

Command	Purpose
show mpls ldp graceful-restart	Displays the LDP graceful-restart configuration.
show mpls ldp neighbor graceful-restart	Displays the graceful-restart parameters for the router's sessions with its LDP neighbors.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS MPLS Command Reference*.

Configuration Examples for MPLS LDP Graceful Restart

The following example shows how to enable MPLS LDP graceful restart, if it has been disabled, in order to preserve the LDP session during an interruption in service:

```
switch# configure terminal
switch(config)# mpls ldp configuration
switch(config-ldp)# graceful-restart
switch(config-ldp)# show mpls ldp graceful-restart neighbor-liveness 200
LDP Graceful Restart is enabled
Neighbor Liveness Timer: 200 seconds
Max Recovery Time: 120 seconds
Forwarding State Holding Time: 600 seconds
Down Neighbor Database (0 records):
Graceful Restart-enabled Sessions:
```

Additional References for MPLS LDP Graceful Restart

For additional information related to implementing MPLS LDP graceful restart, see the following sections:

- Related Documents, page 9-134
- MIBs, page 9-134

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Cisco IOS MPLS LDP graceful restart	MPLS LDP Graceful Restart

MIBs

MIB	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS LDP Graceful Restart

Table 9-2 lists the release history for this feature.

Table 9-2 Feature History for MPLS LDP Graceful Restart

Feature Name	Releases	Feature Information
MPLS LDP graceful restart	5.2(1)	This feature was introduced.

Configuring Basic MPLS TE

This chapter describes how to configure Multiprotocol Label Switching (MPLS) traffic engineering (TE) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS TE

MPLS enabled for traffic engineering makes traditional Layer 2 features available to Layer 3.

This section includes the following topics:

MPLS TE Operation

MPLS TE learns the topology and resources available in a network and then maps traffic flows to particular paths based on resource requirements and network resources such as bandwidth. MPLS TE builds a unidirectional tunnel from a source to a destination in the form of a label switched path (LSP), which is then used to forward traffic. The point where the tunnel begins is called the tunnel headend or tunnel source, and the node where the tunnel ends is called the tunnel tailend or tunnel destination.

MPLS uses extensions to a link-state based Interior Gateway Protocol (IGP), such as Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF). MPLS calculates TE tunnels at the LSP head based on required and available resources (constraint-based routing). If configured, the IGP automatically routes the traffic onto these LSPs. Typically, a packet that crosses the MPLS TE backbone travels on a single LSP that connects the ingress point to the egress point. MPLS TE automatically establishes and maintains the LSPs across the MPLS network by using the Resource Reservation Protocol (RSVP).

MPLS TE is built on the following Cisco NX-OS mechanisms:

- TE tunnel interfaces—From a Layer 2 standpoint, an MPLS TE tunnel interface represents the head of an LSP. It is configured with a set of resource requirements, such as bandwidth, media requirements, and priority. From a Layer 3 standpoint, a TE tunnel interface is the headend of a unidirectional virtual link to the tunnel destination.
- MPLS TE path calculation—This calculation, which operates at the LSP head, determines a path to use for an LSP. The path calculation uses a link-state database that contains flooded topology and resource information.
- Resource Reservation Protocol (RSVP) with TE extensions—RSVP, which operates at each LSP hop, is used to signal and maintain LSPs based on the calculated path.
- MPLS TE link management— Link management, which operates at each LSP hop, performs link call admission on the RSVP signaling messages and tracking of topology and resource information to be flooded.
- Link-state IGP (IS-IS or OSPF)—These IGPs (with TE extensions) globally flood topology and resource information based on link management.
- Enhancements to the SPF calculation used by the link-state IGP (IS-IS or OSPF)—If configured, the IGP automatically routes traffic onto the appropriate TE tunnel based on the tunnel destination. You can also use static routes to direct traffic onto TE tunnels.
- Label switching forwarding—This forwarding mechanism provides routers with a Layer 2-like ability to direct traffic across multiple hops of the LSP established by RSVP signaling.

MPLS TE and HA

MPLS TE supports these Cisco NX-OS high availability (HA) features:

- Nonstop Forwarding (NSF)
- Stateful HA

MPLS TE supports these Cisco NX-OS HA technologies to allow NSF and Stateful HA:

- Stateful Process Restart
- Stateful Switch Over (SSO)

- In-Service Software Upgrade (ISSU)

MPLS TE CSPF Cost Limit

The cost-limit feature allows you to specify the maximum permitted total cost for a tunnel's path. The total cost for a path is the total of the costs of each link traversed. If no path with a total cost less than specified is found, path-calculation fails. The configured cost-limit applies to the metric type that is used while calculating the tunnel's path, which may be the IGP or TE link metrics.

By default, no cost-limit is imposed.

Licensing Requirements for MPLS TE

Product	License Requirement
Cisco NX-OS	MPLS TE feature requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS TE

MPLS TE has the following prerequisites:

- Your network must support Multiprotocol Label Switching (MPLS)
- Your network must support at least one of the following Interior Gateway (IGP) protocols:
  - Intermediate System-to-Intermediate System (IS-IS)
  - Open Shortest Path First (OSPF)
- Ensure that the MPLS feature set is installed.
- Ensure that the MPLS feature set is enabled.

Guidelines and Limitations for MPLS TE

MPLS TE has the following configuration guidelines and limitations:

- MPLS TE supports only a single IGP process or instance. You should not configure MPLS TE in more than one IGP process or instance.
- The IGP process or instance that you configure for MPLS TE must be one of the first four OSPFv2 or IS-IS processes or instances created. Cisco NX-OS Release 6.1 introduces support for more than four process instances for OSPFv2 per VDC. However, only the first four configured OSPFv2 instances are supported with MPLS TE.
- You cannot configure MPLS TE over the logical generic routing encapsulation (GRE) tunnel interface.
- MPLS TE is supported in no more than four VDCs.

Default Settings for MPLS TE

Table 10-1 lists the default settings for basic MPLS TE.

Table 10-1 Default Settings for MPLS TE

Parameters	Default
MPLS TE feature	Disabled

Configuring MPLS TE

This section includes the following topics:

Enabling MPLS TE

You can enable the MPLS TE feature on the device.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature mpls traffic-engineering**
3. (Optional) **show running-config**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**feature mpls traffic-engineering** **Example:** switch(config)# feature mpls traffic-engineering	Enables the MPLS TE feature.
Step 3	**show running-config** **Example:** switch(config)# show running-config	(Optional) Displays information about the running configuration.
Step 4	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring IS-IS for MPLS TE

You can configure IS-IS for MPLS TE.

Note MPLS TE supports a single IGP process or instance. You should not configure MPLS TE in more than one IGP process or instance.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

Note You can configure a router running IS-IS so that Protocol-Independent Multicast (PIM) and MPLS TE can work together with the **mpls traffic-eng multicast-intact** command. You can disable the interoperability between PIM and MPLS TE with the **no mpls traffic-eng multicast-intact** command.

1. **configure terminal**
2. **feature isis**
3. **router isis** *instance-tag*
4. **mpls traffic-eng** {**level-1** | **level-1-2** | **level-2**}
5. **mpls traffic-eng router-id** *interface*
6. (Optional) **show running-config isis**

7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**feature isis** **Example:** switch(config)# feature isis	Enables the IS-IS feature.
Step 3	**router isis** *instance-tag* **Example:** switch(config)# router isis switch(config-router)#	Configures an IS-IS instance and enters router configuration mode. The *instance-tag* can be any case-sensitive, alphanumeric string up to 20 characters.
Step 4	**mpls traffic-eng** {**level-1** | **level-1-2** | **level-2**} **Example:** switch(config-router)# mpls traffic-eng level-1	Configures MPLS TE for IS-IS. You can enable MPLS for level 1, level 2, or level 1 and level 2 routers.
Step 5	**mpls traffic-eng router-id** *interface* **Example:** switch(config-router)# mpls traffic-eng router-id loopback0	Specifies that the TE router identifier for the node is the IP address associated with the configured interface.
Step 6	**show running-config isis** **Example:** switch(config-router)# show running-config isis	(Optional) Displays information about the IS-IS configuration.
Step 7	**copy running-config startup-config** **Example:** switch(config-router)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring OSPF for MPLS TE

You can configure OSPF for MPLS TE.

Note MPLS TE supports a single IGP process or instance. You should not configure MPLS TE in more than one IGP process or instance.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

Note You can configure a router running OSPF so that Protocol-Independent Multicast (PIM) and MPLS TE can work together with the **mpls traffic-eng multicast-intact** command. You can disable the interoperability between PIM and MPLS TE with the **no mpls traffic-eng multicast-intact** command.

1. **configure terminal**
2. **feature ospf**
3. **router ospf** *instance-tag*
4. **mpls traffic-eng area** *area-id*
5. **mpls traffic-eng router-id** *interface*
6. (Optional) **show running-config ospf**
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**feature ospf** **Example:** switch(config)# feature ospf	Enables the IS-IS feature.
Step 3	**router ospf** *instance-tag* **Example:** switch(config)# router ospf 200 switch(config-router)#	Configures an OSPF routing instance and enters router configuration mode. The *instance-tag* can be any case-sensitive, alphanumeric string up to 20 characters.
Step 4	**mpls traffic-eng area** *area-id* **Example:** switch(config-router)# mpls traffic-eng area 1	Turns on MPLS TE for the indicated OSPF area. The *area-id* argument can be an IP address or a positive integer.
Step 5	**mpls traffic-eng router-id** *interface* **Example:** switch(config-router)# mpls traffic-eng router-id loopback0	Specifies that the TE router identifier for the node is the IP address associated with the configured interface.
Step 6	**show running-config ospf** **Example:** switch(config-router)# show running-config ospf	(Optional) Displays information about the OSPF configuration.
Step 7	**copy running-config startup-config** **Example:** switch(config-router)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring MPLS TE on an Interface

You can configure MPLS TE on a TE tunnel egress interface.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **mpls traffic-eng tunnels**
4. **mpls traffic-eng bandwidth** [*interface-kbps* | **percent** *percentage*]

5. **no shut**
6. (Optional) **show interface** *type slot/port*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**interface** *type slot*/*port* **Example:** switch(config)# interface ethernet 2/1 switch(config-if)#	Configures an interface type and enters interface configuration mode. Use **?** to see a list of supported interfaces.
Step 3	**mpls traffic-eng tunnels** **Example:** switch(config-if)# mpls traffic-eng tunnels	Enables MPLS TE tunnels on an interface.
Step 4	**mpls traffic-eng bandwidth** [*interface-kbps* | **percent** *percentage*] **Example:** switch(config-if)# mpls traffic-eng bandwidth 1000	Allocates the MPLS TE bandwidth pool for the interface. The *interface-kbps* argument specifies the maximum amount of bandwidth (in kbps) that may be allocated by TE flows. The range is from 1 to 10000000. The *percentage* argument specifies the maximum percentage of the link bandwidth that may be allocated by TE flows. The range is from 1 to 100.
Step 5	**no shut** **Example:** switch(config-if)# no shut	Activates the interface.
Step 6	**show interface** *type slot*/*port* **Example:** switch(config-if)# show interface ethernet 2/1	(Optional) Displays information about an interface. Use **?** to see a list of supported interfaces.
Step 7	**copy running-config startup-config** **Example:** switch(config-if)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring an MPLS TE Tunnel

You can configure an MPLS TE tunnel with a preferred explicit path or a backup dynamic path option.

Note This configuration applies only to the TE headend node.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **ip unnumbered** *type slot/port*
4. **destination** {*ip-address*}
5. (Optional) **bandwidth** *bandwidth*
6. (Optional) **auto-bw**
7. **path-option** [**protect**] *preference-number* {**dynamic** | **explicit** {**identifier** *id* | **name** *name*} [**verbatim**]} [**lockdown**] [**bandwidth** *kbps*] [**attributes** *listname*]
8. (Optional) **autoroute announce**
9. (Optional) priority
10. **no shutdown**
11. (Optional) **show running-config interface** *int*
12. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1` `switch(config-if-te)#`	Enters TE interface configuration mode. The *number* argument range is from 0 to 65503.
Step 3	**ip unnumbered** *type slot/port* **Example:** `switch(config-if-te)# ip unnumbered loopback 0`	Gives the tunnel interface an IP address that is the same as that of the configured interface. An MPLS TE tunnel interface should use a stable address such as one obtained from a loopback interface. Use **?** to see a list of supported interfaces. **Note** This command is not effective until Lookback0 has been configured with an IP address.
Step 4	**destination** {*ip-address*} **Example:** `switch(config-if-te)# destination 10.3.3.3`	Specifies the destination for a tunnel. The destination must be the MPLS TE router ID of the destination device or the hostname. The *ip-address* is in dotted-decimal notation.
Step 5	**bandwidth** *bandwidth* **Example:** `switch(config-if-te)# bandwidth 250`	(Optional) Configures the bandwidth for the MPLS TE tunnel. The *bandwidth* argument is the bandwidth, in kilobits per second, set for the MPLS TE tunnel. The range is from 1 to 4294967295. The default is 0. If automatic bandwidth is configured for the tunnel, you can use the **bandwidth** command to configure the initial tunnel bandwidth, which will be adjusted by the auto bandwidth mechanism.
Step 6	**auto-bw** **Example:** `switch(config-if-te)# auto-bw`	(Optional) Enables automatic bandwidth changes for the tunnel. You can use the **bandwidth** command to configure the initial tunnel bandwidth, which will be adjusted by the auto bandwidth mechanism.

	Command	Purpose
Step 7	**path-option** [**protect**] *preference-number* {**dynamic** | **explicit** {**identifier** *id* | **name** *name*} [**verbatim**]} [**lockdown**] [**bandwidth** *kbps*] [**attributes** *listname*] **Example:** switch(config-if-te)# path-option 10 explicit name Link5	Configures the tunnel to use a named IP explicit path or a path dynamically calculated from the TE topology database. The *preference-number* range is from 1 to 1000. The *id* range is from 1 to 65535 (with lower numbers preferred). The *name* is any case-sensitive, alphanumeric string. The *kbps* range is from 1 to 4294967295. The *listname* is any case-sensitive, alphanumeric string up to 63 characters. **Note** You can configure multiple path options. TE signals the lowest numbered path option that is valid and meets the constraints. For example, you can specify an explicit path option, and then a less preferred dynamic path option. If the explicit path is not available, then the less preferred dynamic path option is tried.
Step 8	**autoroute announce** **Example:** switch(config-if-te)# autoroute announce	(Optional) Specifies that the IGP should use the tunnel (if the tunnel is up) in its enhanced shortest path first (SPF) calculation.
Step 9	priority Example: switch(config-if-te)# priority	(Optional) Assigns a priority to traffic.
Step 10	**no shutdown** **Example:** switch(config-if-te)# no shutdown	Activates the interface.
Step 11	**show running-config interface** *int* **Example** switch(config-if-te)# show running-config interface tunnel-ts 1	(Optional) Displays information about the interface configuration.
Step 12	**copy running-config startup-config** **Example:** switch(config-if-te)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring Cost Limit

The following are the steps to configure cost limit for an individual TE tunnel:

Step 1 Enter global configuration mode:

switch# **configure terminal**

Step 2 Enter TE interface configuration mode:

switch(config)# **interface tunnel-te** *number*

Step 3 Enter the maximum permitted cost for the tunnel path:

switch(config-if)# **cost-limit** *max-cost*

Configuring an Explicit Path

You can configure an explicit LSP path on the headend router.

Prerequisites

You must have the MPLS TE feature enabled (see "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng configuration**
3. **explicit-path** {**identifier** *id* | **name** *name*}
4. [**index** *number*] {**next-address** [**loose** | **strict**] | **exclude-address**} *address*
5. Repeat step 4 for each router in the path.
6. (Optional) **shutdown**
7. (Optional) **show running-config mpls**
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**mpls traffic-eng configuration** **Example:** switch(config)# mpls traffic-eng configuration switch(config-te)#	Enters MPLS TE configuration mode.
Step 3	**explicit-path** {**identifier** *id* | **name** *name*} **Example:** switch(config-te)# explicit-path name Link5	Enters explicit path configuration mode and creates or modifies the specified path. The *id* range is from 1 to 65535. The *name* is any case-sensitive, alphanumeric string.
Step 4	[**index** *number*] {**next-address** [**loose** | **strict**] | **exclude-address**} *address* **Example:** switch(config-te-expl-path)# index 10 next-address 10.3.3.3	Inserts or modifies a path entry at a specific index. The number range is from 1 to 65535. The *address* represents the node ID and is an IP address in dotted-decimal notation. If you omit the **index** *number* option, multiple command statements are applied in the order in which they are entered. • **Loose** specifies that the previous address (if any) in the explicit path does not need to be directly connected to the next IP address, and that the router is free to determine the path from the previous address (if any) to the next IP address. • **Strict** specifies that the previous address (if any) in the explicit path must be directly connected to the next IP address. • **Exclude-address** excludes an address from subsequent partial path segments. You can enter the IP address of a link or the router ID of a node.
Step 5	Repeat step 4 for each router in the path.	—
Step 6	**shutdown** **Example:** switch(config-te-expl-path)# shutdown	(Optional) Disables the explicit path without deleting the configuration.

	Command	Purpose
Step 7	**show running-config mpls** **Example:** `switch(config-te-expl-path)# show running-config mpls`	(Optional) Displays information about the MPLS configuration.
Step 8	**copy running-config startup-config** **Example:** `switch(config-te-expl-path)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS TE Configuration

To display the MPLS TE configuration, perform one of the following tasks:

Command	Purpose
show mpls traffic-eng	Displays information about MPLS TE.
show mpls traffic-eng tunnels	Displays information about configured MPLS TE tunnels at the head and signaled TE LSPs at other hops.
show run mpls traffic-eng	Displays information about the running configuration of the MPLS TE feature.
show mpls traffic-eng link-management summary	Displays summary information about the MPLS TE link management.
show mpls traffic-eng explicit-paths	Displays information about the MPLS TE explicit paths.
show mpls traffic-eng tunnels brief	Displays brief information about MPLS TE tunnels.
show ip route	Displays the MPLS TE ip route.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Logging Label Switched Path (LSP) Events

Logging helps you monitor your networks. You can configure logging of different events related to tunnels and Label Switched Paths.

All log messages include the following information:

- Name of the tunnel
- Signaled name of the tunnel
- LSP ID of the current LSP

All log messages can be configured on a per-tunnel basis or globally for all TE tunnels. If logging is enabled globally, you cannot disable it for an individual tunnel.

Configuring Tunnel-State Logging

You can configure the generation of syslogs (system messages) when a TE tunnel changes its operational state. A system message is logged to indicate that the tunnel has come up or gone down when either of these events occur. This is in addition to any system message generated by the interface management infrastructure.

No system message is logged if this feature is not configured.

In addition to the information included for all the tunnel log messages, this log contains the new state of the tunnel.

DETAILED STEPS

The following are the steps to configure tunnel-state logging for an individual TE tunnel:

Step 1 Enter global configuration mode:

```
switch# configure terminal
```

Step 2 Enter TE interface configuration mode:

```
switch(config)# interface tunnel-te number
```

Step 3 Configure tunnel state logging:

```
switch(config-te-if)# logging tunnel state
```

The following are the steps to configure tunnel state logging for all the TE tunnels:

Step 1 Enter global configuration mode:

```
switch(config)# configure terminal
```

Step 2 Enter traffic engineering global configuration mode:

```
switch(config)# mpls traffic-eng configuration
```

Step 3 Configure tunnel-state logging:

```
switch(config-te)# logging tunnel state
```

Configuring Tunnel Reoptimization Logging

You can configure a TE tunnel to generate system logs when it is reoptimized successfully. If this feature is configured, and a tunnel is reoptimized, a system message is logged.

Reoptimization messages are not logged under the following conditions:

- Reoptimization Logging feature is not configured
- If a reoptimization attempt does not result in a better path than the current one.
- If a reoptimization is abandoned before completion.

In addition to the information included for all the tunnel log messages, this message includes:

- The ID of the previously used LSP (the LSP that is replaced by reoptimization)

- The reoptimization trigger that caused this attempt.

The following are the steps to configure system logs for an individual tunnel when it attempts reoptimization:

Step 1 Enter global configuration mode:

```
switch# configure terminal
```

Step 2 Enter TE interface configuration mode:

```
switch(config)# interface tunnel-te number
```

Step 3 Configure reoptimization logging:

```
switch(config-te-if)# logging tunnel reoptimize
```

The following are the steps to configure system logs for all tunnels when they are successfully reoptimized:

Step 1 Enter global configuration mode:

```
switch(config)# configure terminal
```

Step 2 Enter traffic engineering global configuration mode:

```
switch(config)# mpls traffic-eng configuration
```

Step 3 Configure re-optimization logging:

```
switch(config-te)# logging tunnel reoptimize
```

Configuring Tunnel Reroute Logging

You can configure a TE tunnel to generate system logs when its reroute-pending state changes. If this feature is configured, and the tunnel has either entered or exited reroute-pending state, a system message is logged.

The reroute pending state bandwidth-change messages are not logged under the following conditions:

- Reroute logging feature is not configured.
- If the tunnel exits the reroute-pending state by going down.

In addition to the information included for all tunnel log messages, this message includes:

- The reason for the tunnel entering reroute-pending state
- The previous LSP's ID (on exit from the reroute-pending state).

The following are the steps to configure system logs for an individual tunnel when its reroute-pending changes:

Step 1 Enter global configuration mode:

```
switch# configure terminal
```

Step 2 Enter TE interface configuration mode:

switch(config)# **interface tunnel-te** *number*

Step 3 Configure reroute-pending change logging:

switch(config-te-if)# **logging tunnel reroute**

The following are the steps to configure system logs for all the tunnels when their reroute-pending changes:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure reroute-pending change logging:

switch(config-te)# **logging tunnel reroute**

Configuring Logging of All the TE Tunnel Events

You can configure generation of system logs for all TE tunnel events other than the ones listed above.

The following are the steps to configure all the system logs for an individual TE tunnel:

Step 1 Enter global configuration mode:

switch# **configure terminal**

Step 2 Enter TE interface configuration mode:

switch(config)# **interface tunnel-te** *number*

Step 3 Configure logging of all system logs:

switch(config-te-if)# **logging tunnel all**

The following are the steps to configure system logs for all tunnels when any event occurs:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure logging all system messages:

switch(config-te)# **logging tunnel all**

Logging Fast Reroute (FRR) Events

You can configure the logging of global messages that are not related to an individual tunnel. All these messages are configured globally. You cannot enable or disable a global message on a per-tunnel or per-interface basis.

Configuring Fast Reroute Backup Assignment Logging

You can configure the generation of system logs when a primary LSP is assigned an FRR backup.

If FRR backup and FRR-ready logging are both configured, the initial backup assignment for a new primary LSP will generate two separate system logs.

The information included in this log is:

- The name and current LSP ID of the backup tunnel.
- The signaled name, source, destination and LSP ID of the protected LSP.
- The type of protection.

The following are the steps to configure FRR backup assignment:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure FRR backup assignment logging:

switch(config-te)# **logging events frr-protection backup**

Configuring Fast Reroute-Ready Logging

You can configure the generation of system logs when a primary LSP moves to the FRR-ready state on assigning a backup tunnel.

A change in the backup tunnel for LSP does not trigger a system log.

The information included in this log are:

- The name and current LSP ISD of the backup tunnel.
- The signaled name, source, destination and LSP ID of the protected LSP.
- The type of protection.

The following are steps to configure FRR ready logging:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure FRR-ready logging:

switch(config-te)# **logging events frr-protection primary ready**

Configuring Fast Reroute-Active Logging

You can configure the generation of system logs when a protected primary LSP transitions to the FRR-active state.

A change in backup tunnel for LSP does not trigger a system log.

The information included in this log are:

- The name and current LSP-id of the backup tunnel.
- The signaled name, source, destination and LSP ID of the protected LSP.
- The type of protection.

The following are the steps to configure FRR-active logging:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure FRR-active logging:

switch(config-te)# **logging events frr-protection primary active**

Configuring All FRR Logging

You can configure the generation of system logs when an FRR event occurs. When configured, a system message is logged to indicate changes to FRR protection.

The information included in this log are:

- The name and current LSP ID of the backup tunnel.
- The signaled-name, source, destination and LSP ID of the protected LSP.
- The type of protection.

The following are the steps to configure all FRR logging:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure logging all events of FRR:

switch(config-te)# **logging events frr-protection all**

Configuring Logging of All Global Events

You can configure the generation of system logs for all non-tunnel TE events.

The following are the steps to configure logging of all nontunnel TE events:

Step 1 Enter global configuration mode:

switch(config)# **configure terminal**

Step 2 Enter traffic engineering global configuration mode:

switch(config)# **mpls traffic-eng configuration**

Step 3 Configure logging of all global events:

switch(config-te)# **logging events all**

Configuration Examples for MPLS TE

This section includes the following configuration examples:

Example: Enabling MPLS TE Using IS-IS

The following example shows how to enable MPLS TE with IS-IS routing:

Enter the following commands on every router or switch in the traffic-engineered portion of your network.

```
feature isis
feature mpls traffic-engineering
router isis 100
 mpls traffic-eng level-1
 mpls traffic-eng router-id loopback0
```

Example: Enabling MPLS TE Using OSPF

The following example shows how to enable MPLS TE with OSPF routing:

Enter the following commands on every router or switch in the traffic-engineered portion of your network.

```
feature ospf
feature mpls traffic-engineering
router ospf 100
 mpls traffic-eng area 0
 mpls traffic-eng router-id loopback0
```

Example: Configuring MPLS TE on an Interface

The following example shows how to configure MPLS TE on an interface:

```
feature mpls traffic-engineering
interface Ethernet 9/1
 mpls traffic-eng tunnels
 mpls traffic-eng bandwidth 1000
 no shut
```

Note The interface must be configured to be used by the IGP. In ISIS, you would have something like the following syntax:

```
ip router isis p1
```

Example: Configuring an MPLS TE Tunnel

The following example shows how to configure a TE tunnel:

```
feature mpls traffic-engineering
interface tunnel-te 1
 ip unnumbered loopback 0
 destination 10.3.3.3
 bandwidth 250
 path-option 10 explicit name Link5
 path-option 20 dynamic
 autoroute announce
 no shut
```

Example: Creating an Explicit Path

The following example shows how to configure an explicit path:

```
feature mpls traffic-engineering
mpls traffic-eng configuration
explicit-path name Link5
 next-address 10.1.1.21
 next-address 10.1.1.10
 next-address 10.1.1.1
 next-address 10.1.1.14
```

Additional References for MPLS TE

For additional information related to implementing MPLS TE, see the following sections:

- Related Document, page 10-158

- MIBs, page 10-158

Related Document

Related Topic	Document Title
MPLS TE commands	*Cisco NX-OS MPLS Command Reference*
MPLS feature set	"Configuring the MPLS Feature Set" chapter

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download Cisco MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: https://cfnng.cisco.com/mibs

Feature Information for MPLS TE

Table 10-2 lists the release history for this feature.

Table 10-2 Feature Information for MPLS TE

Feature Name	Releases	Feature Information
MPLS TE Cost Limit	7.3(0)D1(1)	This feature enables you to specify the maximum permitted total cost for a tunnel's path.
Logging FRR and LSP events	7.3(0)D1(1)	This feature enables you to generate system messages for different events related to TE tunnels and LSPs.
MPLS TE	5.2(1)	This feature was introduced.

Configuring Automatic Bandwidth Adjustment for MPLS TE Tunnels

This chapter describes how to configure automatic bandwidth adjustment for Multiprotocol Label Switching (MPLS) traffic engineering (TE) tunnels on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About Automatic Bandwidth Adjustment for TE Tunnels

The automatic bandwidth adjustment for TE tunnels feature allows you to configure MPLS to automatically monitor and adjust the bandwidth allocation for TE tunnels based on their measured traffic load. The automatic bandwidth behavior changes the configured bandwidth in the running configuration. If automatic bandwidth is configured for a tunnel, TE automatically adjusts the tunnel's bandwidth.

The automatic bandwidth adjustment feature samples the average output rate for each tunnel that is marked for automatic bandwidth adjustment. For each marked tunnel and for the time frequency configured, the feature adjusts the tunnel's allocated bandwidth to be the largest sample for the tunnel since the last adjustment.

The frequency with which the tunnel bandwidth is adjusted and the allowable range of adjustments is configurable on a per-tunnel basis. The sampling interval and the interval over which to average the tunnel traffic to obtain the average output rate are user configurable on a per-tunnel basis.

This feature adjusts the TE tunnel's bandwidth, which is the amount of bandwidth requested by a tunnel. Tunnels only use linked with enough bandwidth left to accommodate this request. (TE link bandwidth is the pool of bandwidth from which TE tunnels allocate requested amount. Tunnels can traverse only those links with enough bandwidth left to satisfy the requirement.)

Licensing Requirements for Automatic Bandwidth Adjustment for TE Tunnels

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Automatic bandwidth adjustment for TE tunnels requires an MPLS license. For a complete explanation of the NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for Automatic Bandwidth Adjustment for TE Tunnels

The automatic bandwidth adjustment for TE tunnels feature has the following prerequisites:

- You must enable the MPLS TE feature. MPLS TE can be enabled or disabled by the [**no**] **feature mpls traffic-eng** command and is enabled by default.
- You must configure the MPLS TE feature by using the **mpls traffic-eng bandwidth** command on each link that a tunnel crosses.

  If the **bandwidth** command is configured for the tunnel, the command configures the initial tunnel bandwidth, which is adjusted by the automatic bandwidth operation.

Note If you configure a tunnel's bandwidth with the **bandwidth** command and configure the minimum amount of automatic bandwidth with the **auto-bw** command, the minimum amount of automatic bandwidth adjustment is the lower of those two configured values.

Guidelines and Limitations for Automatic Bandwidth Adjustment for TE Tunnels

The automatic bandwidth adjustment feature has the following configuration guidelines and limitations:

- The automatic bandwidth adjustment feature adjusts the bandwidth for each tunnel according to the adjustment frequency configured for the tunnel and the sampled output rate for the tunnel since the last adjustment. The adjustment feature does not consider any adjustments previously made or pending for other tunnels.
- If a tunnel is brought down to calculate a new label switched path (LSP) because the LSP is not operational, the configured bandwidth is removed. If the router is reloaded, the system gives a new configured bandwidth.
- You cannot configure MPLS TE over the logical generic routing encapsulation (GRE) tunnel interface.
- MPLS traffic engineering should not be configure in more than one IGP process/instance.

Default Settings for Automatic Bandwidth Adjustment for TE Tunnels

Table 11-1 lists the default settings for automatic bandwidth adjustment for TE tunnels.

Table 11-1 Default Settings for Automatic Bandwidth Adjustment for TE Tunnels

Parameters	Default
Frequency	86400 seconds

Configuring Automatic Bandwidth Adjustment for TE Tunnels

This section includes these topics:

Enabling Automatic Bandwidth Adjustment on a Platform

You can enable automatic bandwidth adjustment on a platform and initiate sampling of the output rate for tunnels that are configured for bandwidth adjustment.

Note This task applies only to the TE headend router. The configuration applies to all locally configured TE headend interfaces.

Prerequisites

You must enable the MPLS TE feature (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng**
3. **auto-bw timers** [**frequency** *seconds*]
4. **no auto-bw timers**
5. **end**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls traffic-eng** **Example:** `switch(config)# mpls traffic-eng` `switch(config-te)#`	Enters traffic engineering global configuration mode.
Step 3	**auto-bw timers** [**frequency** *seconds*] **Example:** `switch(config-te)# auto-bw timers frequency 300`	Enables automatic bandwidth adjustment on a platform and begins sampling the output rate for tunnels that have been configured for automatic bandwidth adjustment. The *seconds* argument specifies the interval, in seconds, for sampling the output rate of each tunnel configured for the automatic bandwidth adjustment. The range is from 1 through 604800. The recommended value is 300.
Step 4	**no auto-bw timers** **Example:** `switch(config-te)# no auto-bw timers`	(Optional) Disables the automatic bandwidth adjustment on a platform. Use the **no** version of the command, which terminates the output rate sampling and bandwidth adjustment for tunnels. In addition, the **no** form of the command restores the configured bandwidth for each tunnel where the configured bandwidth is determined as follows: • If the tunnel bandwidth was explicitly configured with the **bandwidth** command after the running configuration was written to the startup configuration, the configured bandwidth is the bandwidth specified by that command. • If the tunnel bandwidth was not explicitly configured with the **bandwidth** command, the configured bandwidth is the bandwidth specified for the tunnel in the startup configuration.
Step 5	**end** **Example:** `switch(config-te)# end` `switch#`	Exits to EXEC mode.

Enabling Automatic Bandwidth Adjustment for a TE Tunnel

You can enable the automatic bandwidth adjustment for a tunnel and specify the range of automatic bandwidth adjustments applied to the tunnel.

Tip Each **auto-bw** command supersedes the previous one. To specify multiple options for a tunnel, you must specify them all in a single **auto-bw** command.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **auto-bw [collect-bw] [[frequency** *seconds***] [min-bw** *kbps***] [max-bw** *kbps***]]**
4. **end**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1` `switch(config-if-te)#`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**auto-bw [collect-bw] [[frequency** ***seconds***`]` **[min-bw** ***kbps***`] [max-bw` ***kbps***`]]` **Example:** `switch(config-if-te)# auto-bw max-bw 2000` `min-bw 1000 frequency 300`	Enables the automatic bandwidth adjustment for the tunnel and controls the manner in which the bandwidth for a tunnel is adjusted. **NOTE**: The **collect-bw** and **min-bw/max-bw** variables are mutually exclusive because the switch does not perform an actual application when you specify **collect-bw.**
Step 4	**end** **Example:** `switch(config-if-te)# end` `switch#`	Exits to EXEC mode.

Verifying the Automatic Bandwidth Configuration

To verify the automatic bandwidth configuration, perform one of the following tasks:

Command	Purpose
show mpls traffic-eng tunnels	Displays information about tunnels. It includes automatic bandwidth information for tunnels that have the feature enabled.
show running-config interface tunnel-te *id*	Verifies that the **tunnel mpls traffic-eng auto bw** command is set appropriately.

The following example shows how to display information about tunnels. The command output shows the following:

- The auto-bw line indicates that the automatic bandwidth adjustment is enabled for the tunnel.
- 86400 is the time, in seconds, between bandwidth adjustments.

- 86258 is the time, in seconds, remaining until the next bandwidth adjustment.
- 0 is the largest bandwidth sample since the last bandwidth adjustment.
- 0 is the last bandwidth adjustment and the bandwidth currently requested for the tunnel.

```
switch# show mpls traffic-eng tunnels tunnel-te 2

Name: N7K-Get-well-R1_t2                (tunnel-te2) Destination: 10.0.0.4
Status:
     Admin: up         Oper: up     Path: valid       Signalling: connected
     path option 1, type explicit path2(Basis for Setup, path weight 80)

   Config Parameters:
     Bandwidth: 500        kbps (Global)  Priority: 7  7   Affinity: 0x0/0xffff
     Metric Type: TE (default)
     AutoRoute: disabled LockDown: disabled
     auto-bw: (300/245) 583 Bandwidth Requested: 555
           Samples Missed 1: Samples Collected 1
   Active Path Option Parameters:
     State: explicit path option 1 is active
     BandwidthOverride: disabled  LockDown: disabled  Verbatim: disabled

InLabel  :  -
   OutLabel : Ethernet1/26, 2003
     RSVP Signalling Info:
          Src 10.0.0.2, Dst 10.0.0.4, Tun_Id 2, Tun_Instance 2
     RSVP Path Info:
       My Address: 10.0.0.2
       Explicit Route: 24.0.0.2.26.0.0.1.26.0.0.2 10.0.0.4
       Record   Route: NONE
       Tspec: ave rate=555 kbits, burst=1000 bytes, peak rate=555 kbits
     RSVP Resv Info:
       Record   Route: 24.0.0.2 26.0.0.2
       Fspec: ave rate=555 kbits, burst=1000 bytes, peak rate=555 kbits
   Shortest Unconstrained Path Info:
     Path Weight: 80 (TE)
     Explicit Route: 22.0.0.1 22.0.0.2 25.0.0.1 25.0.0.2
                     10.0.0.4
   History:
     Tunnel:
       Time since created: 7 minutes, 43 seconds
       Time since path change: 2 minutes, 21 seconds
       Number of LSP IDs (Tun_Instances) used: 2
     Current LSP:
       Uptime: 1 minutes, 23 seconds
       Selection: reoptimization
     Prior LSP:
       ID: pat option 1 [1]
       Removal trigger: configuration changed
```

The following example shows how to verify that the **tunnel mpls traffic-eng auto bw** command is set appropriately. The command output shows that the bandwidth value has changed after adjustment (the **bandwidth is 1500**).

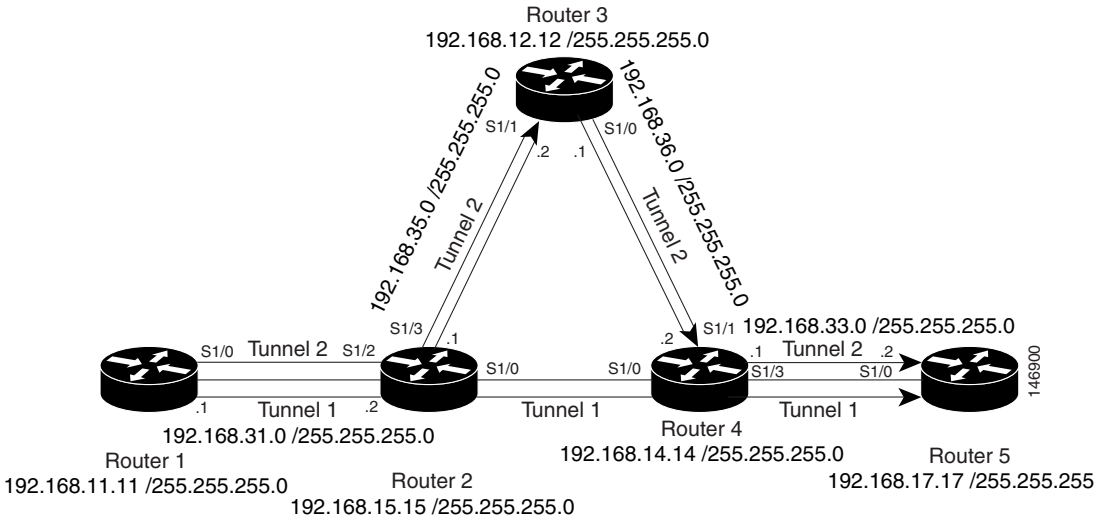
```
switch# show running-config interface tunnel-te1
!Time: Mon Nov 25 19:32:35 2013
.
version 6.2(6)
.
interface tunnel-te1
  ip unnumbered loopback0
  no shutdown
  destination 10.0.0.4
```

```
 auto-bw frequency 300 min-bw 300
 autoroute announce
 bandwidth 583
 path-option 1 explicit name path1
```
record-route

Configuration Examples for Automatic Bandwidth Adjustment for TE Tunnels

Figure 11-1 shows a sample MPLS topology. The following sections contain sample configuration examples about configuring an automatic bandwidth adjustment for MPLS TE tunnels that originate on Router 1 and enabling automatic bandwidth adjustment for Tunnel 1.

Figure 11-1 Sample MPLS Traffic Engineering Tunnel Configuration



This section provides the following configuration examples based on Figure 11-1:

- Example: Configuring the MPLS Traffic Engineering Automatic Bandwidth, page 11-167
- Example: Tunnel Configuration for Automatic Bandwidth, page 11-168

The examples omit some configuration required for MPLS TE, such as the required Resource Reservation Protocol (RSVP) and Interior Gateway Protocol (IGP), and either Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF) configuration. The examples show the configuration for automatic bandwidth adjustment.

Example: Configuring the MPLS Traffic Engineering Automatic Bandwidth

The following example shows how to use the **auto-bw timers** command to enable an automatic bandwidth adjustment for Router 1. The command specifies that the output rate is to be sampled every 10 minutes for tunnels configured for automatic bandwidth adjustment.

```
configure terminal
!
mpls traffic-eng
```

```
 auto-bw timers frequency 600  !Enable automatic bandwidth adjustment
interface loopback 0
 ip address 192.168.11.11 255.255.255.0
```

Example: Tunnel Configuration for Automatic Bandwidth

The following example shows how to use the **auto-bw** command to enable an automatic bandwidth adjustment for Tunnel 1. The command specifies a maximum allowable bandwidth of 2000 kbps, a minimum allowable bandwidth of 1000 kbps, and a default automatic bandwidth adjustment frequency of once a day.

```
interface tunnel-te1
 ip unnumbered loopback 0
 destination 192.168.17.17
 bandwidth 1500
 priority 1 1
 path-option 1 dynamic
 auto-bw max-bw 2000 min-bw 1000   !Enable automatic bandwidth
                                   !adjustment for Tunnel1
```

Additional References

The following sections provide references related to the automatic bandwidth adjustment for TE tunnels feature.

Related Documents

Related Topic	Document Title
IS-IS and OSPF commands	*Cisco NX-OS Unicast Routing Command Reference*
MPLS commands	*Cisco NX-OS Multiprotocol Label Switching Command Reference*
Quality of service commands	*Cisco NX-OS Quality of Service Commands*
Quality of service solutions configuration	*Quality of Service Overview*

Standards

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

MIBs

MIB	MIBs Link
MPLS Traffic Engineering MIB	To locate and download MIBs for selected platforms, Cisco NX-OS releases, and feature sets, use Cisco MIB Locator found at the following URL: https://cfnng.cisco.com/mibs

RFCs

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

Feature History for Automatic Bandwidth Adjustment for TE Tunnels

Table 11-2 lists the release history for this feature.

Table 11-2 Feature History for MPLS Traffic Engineering—Automatic Bandwidth Adjustment for TE Tunnels

Feature Name	Releases	Feature Information
Automatic Bandwidth Adjustment for TE Tunnels	6.2(6)	This feature was introduced.

Configuring MPLS RSVP TE

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Resource Reservation Protocol (RSVP) Traffic Engineering (TE) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

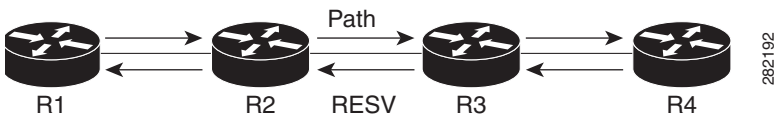
Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS RSVP TE

RSVP is a signaling protocol that reserves resources, such as for IP unicast and multicast flows, and requests quality-of-service (QoS) parameters for applications. The protocol was extended in MPLS RSVP TE to enable RSVP to set up label switched paths (LSPs) that can be used for TE in MPLS networks.

Figure 12-1 shows how RSVP sets up an LSP from router R1 through router R4 that can be used for TE in an MPLS environment.

Figure 12-1 Example of RSVP Used to Set Up an MPLS LSP



The LSP setup is driven by the TE application on the headend router R1 and is identified as a session that specifies the tailend router for the LSP (R4), a tunnel identifier, and an extended tunnel identifier, which is typically the local address of R1.

The headend RSVP component signals a PATH message destined toward R4. The PATH message can include policy link-admission control information, which identifies the sender that is setting up the path, and a flow specification that defines the resources desired on the path.

Each hop along the path examines the PATH message, verifies the policy control information, saves the path state that is associated with the session, and sets aside the requested resources specified by the sender. When the tailend router is reached, a hop-by-hop reservation (RESV) message is initiated by R4 toward R1, along the reverse direction taken by the PATH message.

At each node including the tailend, the session-state is updated, the earmarked resources are reserved for the session, and an MPLS label is allocated for use by the prior hop. When the RESV reaches the headend router, the LSP setup for the session is complete.

The reservation state in each router is considered as a soft state, which means that periodic PATH and RESV messages must be sent at each hop to maintain the state. If there is a failure to establish or maintain a session at any hop, RSVP provides messages to propagate the error along the path, and to tear down the existing reservation.

Overview

RSVP interacts with TE to support the MPLS TE functionality.

The TE process contains the following functionalities:

- End-point control, which is associated with establishing and managing TE tunnels at the headend and tailend.
- Link-management, which manages link resources to do resource-aware routing of TE LSPs and to program MPLS labels.
- Fast Reroute (FRR), which manages the LSPs that need protection and to assign backup tunnel information to these LSPs.

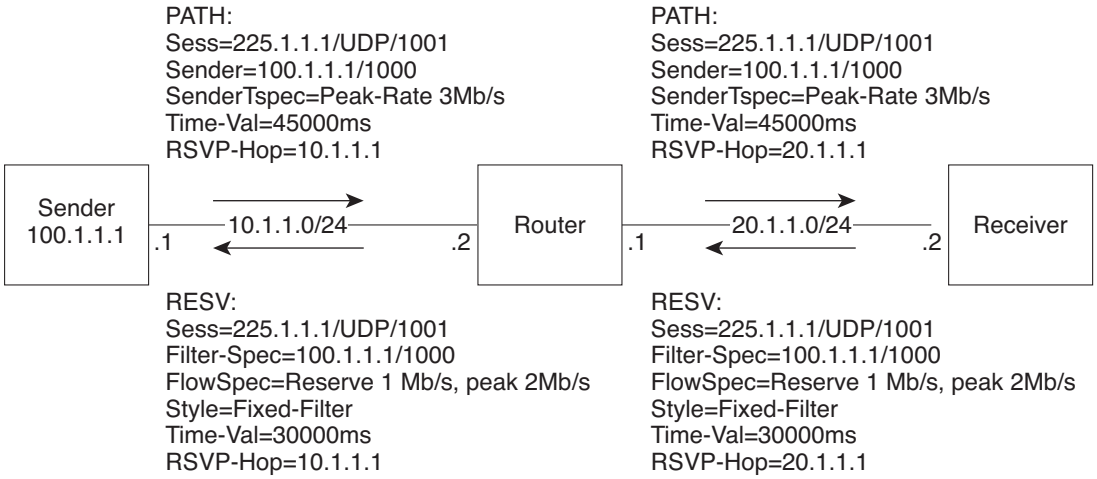
Note The interactions between TE and RSVP assume the existence of the end-point control, link-management, and FRR functionality within TE.

RSVP Core Functionality

The RSVP core functionality specifies RSVP messages and the objects required to set up resource reservations for IP unicast and multicast flows.

The primary RSVP messages are PATH and RESV messages. Senders send a PATH message from the source to the receiver to specify the reservation requirements of a data flow. Receivers send a RESV message to the sender to reserve resources for the flow. The primary object is the Session object, which identifies the data flow via the destination address, IP protocol ID, and destination port. Other key objects include the Sender-Template and Sender-Tspec, which the switch uses to qualify the sender and traffic specification in the PATH message, and the Filter-Spec, FlowSpec, and Style, which the switch uses in the RESV message to further classify the flow, specify its resource requirements, and designate the reservation style. An example that shows a resource reservation for the {225.1.1.1, UDP, 1001} data flow through the PATH and RESV messages and their objects is shown in Figure 12-2. The figure also shows the Time-Val and RSVP-Hop objects that track the PATH refresh and previous hop in the PATH message and the RESV refresh and next hop in the RESV message.

Figure 12-2 RSVP Reservation via PATH and RESV Messages



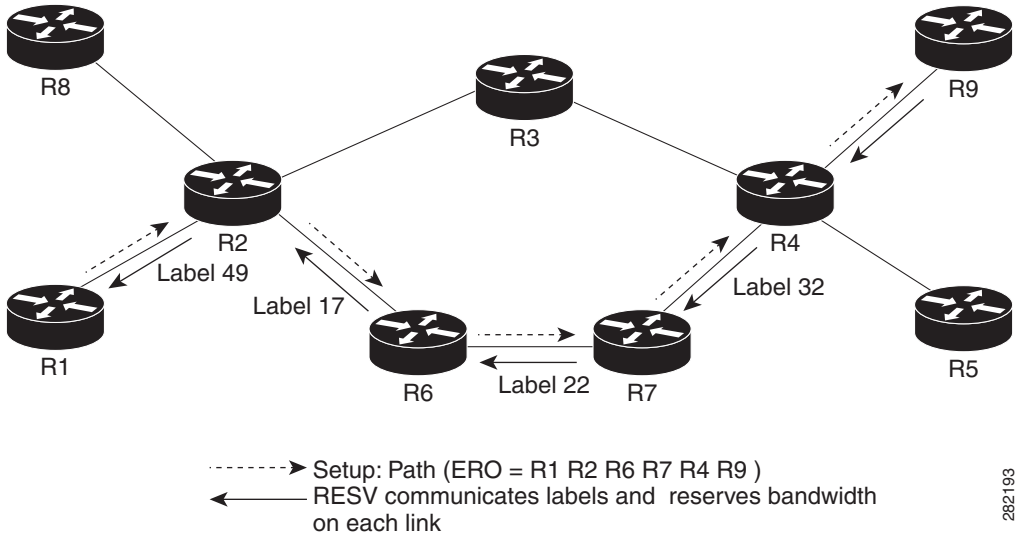
In addition to the PATH and RESV messages, RSVP also specifies support messages that include PATH-ERROR, RESV-ERROR, PATH-TEAR, RESV-TEAR, and RESV-CONFIRM messages that are used to handle error situations, to tear down existing reservations, and to confirm the setup of an existing reservation. Other message objects include an Integrity object that provides integrity protection to RSVP messages, a Policy-Data object that identifies the sender and receiver credentials, a Scope object that carries an explicit list of senders to which a RESV is to be sent, an Error-Spec object that provides error information, a Resv-Confirm object that identifies the receiver interested in the confirmation message, and an Adspec object that carries flow advertisement information.

RSVP TE (RFC 3209, 5151)

RSVP TE builds on the RSVP core protocol, defines new objects, and modifies existing objects used in the PATH and RESV objects for LSP establishment. The base *Session* construct for RSVP TE is based on the triple {Tunnel Remote Address, Tunnel ID, Extended Tunnel ID}. The *Sender Template* object contains the {IPv4 tunnel sender address, LSP-ID}. The PATH message was extended to contain a *Label-Request* object (LRO) that results in a label being assigned during the RESV, a *Session-Attribute* object that is used to provide additional requirements for a session, and an *Explicit-Route* object (ERO) that specifies the data path traversed by the PATH message, which could be independent of IP-routing. The RESV message was modified to include a *Label* object that contains the MPLS label and a *Record-Route* object (RRO) to record the path taken followed by the RESV message. The *Flowspec* object was also modified to set up a reservation on LSPs.

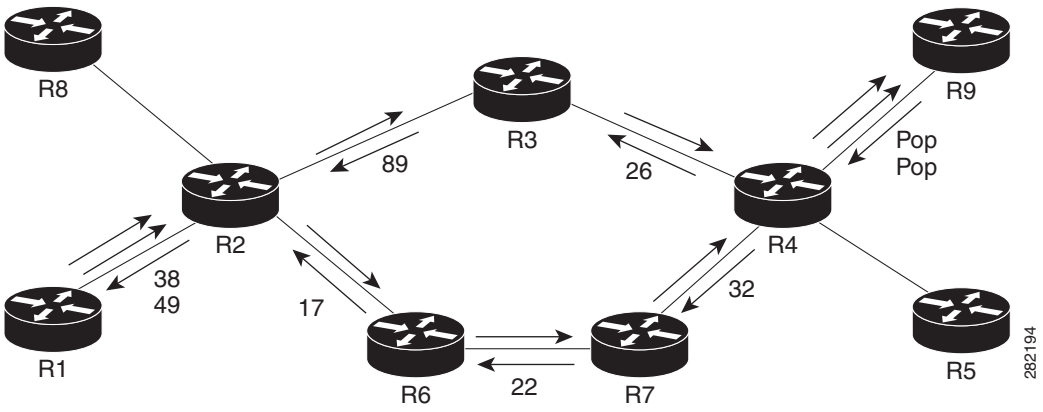
An example of an LSP path setup that uses procedures described in RSVP TE is shown in Figure 12-3. The figure shows a PATH message that is sent from router R1 to Router R9 with an *ERO* object of R1-R2-R6-R7-R4-R9. The PATH message contains a *LRO* object in the message. The RESV message shows labels that are assigned in the reverse direction of the PATH message, which is done via the *Label* object. The LSP setup is driven by the TE process on the router.

Figure 12-3 MPLS LSP Setup Using RSVP TE



For RSVP TE path reoptimization, the switch must reroute an LSP to a new data path before destroying the existing LSP, which is known as a make-before-break. Path reoptimization, as shown in Figure 12-4, is achieved when the switch sends two PATH messages with the same session ID but with different sender templates. When the receiver receives this PATH message, it recognizes that make-before-break is in progress and sends a RESV to reserve resources using the shared-explicit reservation *Style* object, which allows for sharing the resource requirements of the two paths before the original path is torn down.

Figure 12-4 TE Path Reoptimization Using RSVP



RSVP TE Explicit Routing (Strict, Loose)

RSVP TE explicit routes are particular paths in the network topology that you can specify as abstract nodes, which could be a sequence of IP prefixes or a sequence of autonomous systems, in the ERO. The explicit path could be administratively specified, or automatically computed using an algorithm such as constrained shortest path first (CSPF).

The explicit path specified in the ERO may be a strict path or a loose path.

A strict path means that a network node and its preceding node in the ERO must be adjacent and directly connected.

A loose hop means that a network node specified in the ERO must be in the path but is not required to be directly connected to its preceding node. If a loose hop is encountered during ERO processing, the node that processes the loose hop can update the ERO with one or more nodes along the path from itself to the next node in the ERO. The advantage of a loose path is that the entire path does not need to be specified or known when creating the ERO. The disadvantage of a loose path is that it can result in forwarding loops during transients in the underlying routing protocol.

RSVP Hello

The RSVP hello functionality was introduced in RFC 3209 to monitor communication failures with a neighbor. A hello message can contain a *Hello-Request* object or a *Hello-Ack* object. The hello message is periodic and unidirectional, so that each neighbor can issue a Hello-Request with a period that is independent of its neighbor, and a Hello-Ack must also be sent in response to the Hello-Request. The Hello-Request includes a source-instance that is echoed by the neighbor in its Hello-Ack. The Hello-Request also includes a destination-instance that echoes the destination-instance value used by the neighbor in its Hello-Ack. Instance values are fixed during a session and must be changed when communication breaks down, which enables a router to identify a communication failure with its neighbor, apart from a hello timeout.

RSVP Fast Reroute

When a router's link or neighboring node fails, the router often detects this failure by receiving an interface-down notification. When a router notices that an interface has gone down, it switches LSPs going out that interface onto their respective backup tunnels (if any).

RSVP establishes backup LSP-based tunnels for the local repair of TE LSPs. RSVP uses the facility backup method in which a PLR creates one or more bypass tunnels that can be used to protect multiple LSPs.

The Fast-Reroute object is used in the PATH message and contains a flag that identifies the backup method to be used as facility-backup. The Fast-Reroute object specifies setup and hold priorities, which are included in a set of attribute filters and bandwidth requirements to be used in the selection of the backup path.

The Session-Attribute object signals in the PATH message that local protection is desired for an LSP, requires label recording when doing a record route, explicitly specifies a desire for node or bandwidth protection on an LSP, and specifies the use of a shared-explicit style of a reservation by the egress node.

The *RRO* object reports in the RESV message the availability or use of local protection on an LSP, and whether bandwidth and node protection are available for that LSP.

The signaling of the FRR requirements is initiated at the TE tunnel headend. PLRs along the path act on the FRR requirements based on the backup tunnel availability at the PLR, and signal the backup tunnel selection information to the headend. When an FRR event is triggered, the PLR sends PATH messages through the backup tunnel to the merge point (MP) where the backup tunnel rejoins the original LSP. The MP also sends RESV messages to the PLR using the RSVP-Hop object that is included by the PLR in its PATH message. This process prevents the original LSP from being torn down by the MP. Also, the PLR signals the tunnel headend with a PATH-ERROR message to indicate the failure along the LSP and that FRR is in active use for that LSP. This information is used by the headend to signal a new LSP for the TE tunnel, and to tear down the existing failed path after the new LSP is set up through make-before-break techniques.

You can detect FRR link failures by monitoring the interface states that are configured for the links or by using Bidirectional Forwarding Detection (BFD) to detect the health of the neighbor.

Refresh Reduction

RSVP requires that the path and reservation state that are set up during LSP signaling must be refreshed by periodically sending PATH refresh and RESV refresh messages. Refresh messages are used to synchronize the state between RSVP neighbors and to recover from lost RSVP messages. Periodic refresh signaling, however, can result in scaling issues and reliability and latency issues depending on the refresh period.

A refresh-reduction-capable-bit in the RSVP message common header is set by nodes to indicate support for refresh reduction capability. A Message-ID extension defines three new objects, a Message-ID object, a Message-ID-Ack object, and a Message-ID-Nack object. The Message-ID object contains a sender selected value, which when combined with the sender's IP address identifies the particular RSVP message and the state that it represents.

Reliable Messages

To support a reliable RSVP message exchange, a sender includes the Message-ID object in the RSVP message and sets an Ack-Desired flag in the Message-ID object to indicate that it wants an explicit acknowledgement of the message from the neighbor. The sender retransmits unacknowledged messages at a rate faster than the standard refresh period, until a retry limit is reached. The Message-ID-Ack object is used by a receiver to acknowledge receipt of the messages. The object can be sent in an explicit Ack message to the receiver, or it can be combined with another RSVP message if the message is ready to be sent to the sender.

To reduce the volume of the PATH and RESV refresh messages, RSVP-Refresh-Red defines a summary refresh (Srefresh) message that can be sent between RSVP neighbors. The Srefresh message contains a summary refresh extension that carries a list of Message-ID objects that correspond to PATH and RESV messages that originally established the PATH and RESV state. An RSVP node that receives an Srefresh message matches each listed Message-ID object with the installed PATH and RESV state and updates the state as if a normal RSVP refresh message has been received. If the matching state is not found, the Srefresh sender is notified through an Ack message or a combined Ack message, that contains a Message-ID-Nack object.

Message Authentication

RSVP enables particular users to obtain preferential access to network resources under the control of an admission control function that protects RSVP message integrity hop-by-hop by transmitting an authenticating digest of the message prepared using a shared secret authentication key, a sequence number, and a keyed-hash algorithm in the Integrity object of the RSVP message. This process allows the message receiver to identify playbacks and to thwart replay attacks. The scheme may also use an RSVP Integrity Challenge and response messages with a Challenge object to initialize the sequence number used between the sender and receiver The keyed-hash algorithms that are supported in this project are HMAC-MD5 and HMAC-SHA1.

Note For authentication to work properly, two connected switches must be configured with authentication. You must also synchronize the keychain (key-id/key-strings) configuration between the switches participating in the authenticated exchange.

The **authentication** [**neighbor address** *IP-address*] **key-chain** *key-chain-name* command is used to authenticate the neighbor.

Note You must configure keychain parameters before the **authentication** commands can take effect.

```
switch# configure terminal
switch(config)# key chain key1
switch(config-keychain)# key 4660
switch(config-keychain-key)# key-string qwertyui
==============Now configuring Auth for RSVP===========
switch(config)# ip rsvp
switch(config-ip-rsvp)# authentication key key1
switch(config-ip-rsvp)#

switch(config)# ip rsvp
switch(config-ip-rsvp)# authentication key-chain key1

switch(config)# feature mpls traffic-engineering
switch(config)# interface eth 2/1
switch(config-if)# ip rsvp authentication lifetime 20:30:30


[no] authentication [neighbor address <IP-address>] key-chain <key-chain-name>
[no] authentication [neighbor address <IP-address>] type {md5 | sha-1}
[no] authentication [neighbor address <IP-address>] lifetime <hh:mm:ss>
[no] authentication [neighbor address <IP-address>] window-size  <value>
[no] authentication [neighbor address <IP-address>] challenge

[no] ip rsvp authentication key-chain <key-chain-name>
[no] ip rsvp authentication type {md5 | sha-1}
[no] ip rsvp authentication lifetime <hh:mm:ss>
[no] ip rsvp authentication window-size  <value>
```

These sets of commands are completely independent and depend on the type of authentication required. For example, if you want to globally configure authentication for RSVP, you would use the following set of commands.

Note The optional **neighbor address** keyword and argument is not present.

```
switch(config)# ip rsvp
switch(config-ip-rsvp)# authentication key-chain key-chain-name
switch(config-ip-rsvp)# authentication type {md5 | sha-1}
switch(config-ip-rsvp)# authentication lifetime <hh:mm:ss>
switch(config-ip-rsvp)# authentication window-size  <value>
switch(config-ip-rsvp)# authentication challenge
```

```
[no] authentication key-chain <key-chain-name>
[no] authentication type {md5 | sha-1}
[no] authentication lifetime <hh:mm:ss>
[no] authentication window-size  <value>
[no] authentication challenge
```

If a per-interface authentication is needed, that is, if all RSVP neighbors on the other side of the interface are to be authenticated, then an interface set of commands is used:

```
[no] ip rsvp authentication key-chain <key-chain-name>
[no] ip rsvp authentication type {md5 | sha-1}
[no] ip rsvp authentication lifetime <hh:mm:ss>
[no] ip rsvp authentication window-size  <value>
```

If a user has only some specific neighbors that require to be authenticated then this is a set to be used.

Note The **neighbor** keyword and argument is no longer optional:

```
switch(config)# ip rsvp
switch(config-ip-rsvp)# authentication neighbor address <IP-address> key-chain
<key-chain-name>
switch(config-ip-rsvp)# authentication neighbor address <IP-address> type {md5 | sha-1}
switch(config-ip-rsvp)# authentication neighbor address <IP-address> lifetime <hh:mm:ss>
switch(config-ip-rsvp)# authentication neighbor address <IP-address> window-size  <value>
switch(config-ip-rsvp)# authentication neighbor address <IP-address> challenge
```

```
[no] authentication neighbor address <IP-address> key-chain <key-chain-name>
[no] authentication neighbor address <IP-address> type {md5 | sha-1}
[no] authentication neighbor address <IP-address> lifetime <hh:mm:ss>
[no] authentication neighbor address <IP-address> window-size  <value>
[no] authentication neighbor address <IP-address> challenge
```

To verify that this command has been configured correctly, use the **show ip rsvp authentication** command.

RSVP Bundle Messages

RSVP bundle messages consist of a bundle header followed by a variable number of standard RSVP messages. The bundle message aggregates multiple RSVP messages within a single PDU, though they can only be sent to RSVP neighbors that support bundling. The maximum size of an RSVP message is one IP datagram.

```
switch(config-ip-rsvp)# signalling refresh reduction bundle-max-size
```

Graceful Restart

The RSVP graceful restart (GR) is based on using RSVP hellos and adds new objects; for example, a Restart-Capability object to the Hello message and a Recovery-Label object to the sender template that forms part of the PATH message. The RSVP graceful restart procedure also adds a new message; for example, the RECOVERY PATH message has been added to help the restart.

An example that depicts the RSVP GR functionality is shown in Figure 12-5. The figure shows a TE tunnel between R1 and R3 and RSVP hellos being exchanged between routers R1 and R2, and R2 and R3 respectively. The Hello message contains a Restart-Capability object that defines a restart time and a recovery time for the router that originated the message. If router R2 restarts, the neighboring routers R1 and R3 detect the failure when four Hello messages with the Hello-Ack object are not received from R2. These routers start a restart timer based on the restart time specified by R2 in its Restart-Capability object. If R2 fails to restart during this restart-period, R1 and R3 tearing down the existing TE LSP between R1 and R4.

Figure 12-5 RSVP Graceful Restart Example

If R2 restarts during the restart period, it sends a Hello message to R1 and R3, which check the source-instance in the Hello against the value used by R2 before the failure.

If the value is unchanged, then R1 and R3 associate the R2 failure with a control channel failure and send summary refresh messages to R2 to refresh the state. Otherwise, R1 and R3 assume that R2 has restarted and check the recovery time sent by R2 in its HELLO *Restart-Capability* object.

If the recovery time is set to 0, R1 and R3 assume that R2 was not able to preserve its forwarding state, otherwise the value specified by R2 is considered as the recovery period for R2. During the recovery period, R1 sends PATH messages to R2 with Recovery-Label objects that contain labels previously sent by R2 to R1. This process enables R2 to recover the labels that it has given to R1 and locate the corresponding outgoing label and interface for the LSP (for example from TE). R2 then sends a corresponding PATH message to R3 with a Suggested-Label object. On receipt of the PATH message, R3 sends a RESV to R2.

If the downstream router R3 passes a label that does not match the label in the Suggested-Label object, R2 must reconfigure itself to use this label or generate a RESV error. R2 should not send data upstream using the label in the Suggested-Label object until the downstream router passes a label to R2 in its RESV messages.

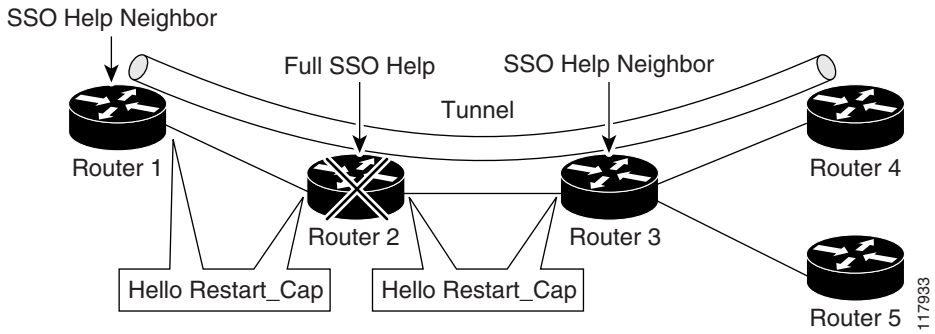
Overview of MPLS TE and RSVP Graceful Restart

RSVP graceful restart allows TE RSVP-enabled nodes to recover gracefully after a node failure in the network so that the RSVP state after the failure is restored as quickly as possible. The node failure can be completely transparent to other nodes in the network.

RSVP graceful restart preserves the label values and forwarding information and works with third-party or Cisco routers seamlessly.

RSVP graceful restart depends on RSVP hello messages to detect that a neighbor went down. Hello messages include Hello Request or Hello Acknowledgment (ACK) objects between two neighbors. As shown in Figure 12-6, the RSVP graceful restart extension to these messages adds an object called Hello Restart_Cap, which tells neighbors that a node may be able to reconnect if a failure occurs.

Figure 12-6 How RSVP Graceful Restart Works



The Hello Restart_Cap object has two values: the restart time, which is the sender's time to restart the RSVP_TE component and exchange hello messages after a failure; and the recovery time, which is the desired time that the sender wants the receiver to synchronize with the RSVP and MPLS databases.

In Figure 12-6, RSVP graceful restart help neighbor support is enabled on routers 1 and 3 so that they can help a neighbor recover after a failure, but they cannot do self recovery. Router 2 has full SSO help support enabled, which means that it can do self recovery after a failure or help its neighbor to recover. Router 2 has two RPs, one that is active and one that is standby (backup). A TE LSP is signaled from router 1 to router 4.

Router 2 does checkpointing; it copies state information from the active RP to the standby RP, which ensures that the standby RP has the latest information. If an active RP fails, the standby RP can take over.

Routers 2 and 3 exchange periodic graceful restart hello messages every 10,000 milliseconds (ms) (10 seconds), and so do routers 2 and 1 and routers 3 and 4. Assume that router 2 advertises its restart time = 60,000 ms (60 seconds) and its recovery time = 60,000 ms (60 seconds) as shown in the following example:

The debug from the sender side is as follows:

```
2011 Mar 15 11:47:46.200375 rsvp: [3016] HELLO-MSG: RSVP-HELLO: Sending msg_type [1=Req,
2=Ack] 1 from 10.1.1.1 to 10.1.1.2
```

The receiver debugs are as follows:

```
2011 Mar 15 11:48:26.012252 rsvp: [2967] HELLO-MSG: Received HELLO msg with object length
24 on i/f 10.21.1.2
2011 Mar 15 11:48:26.012344 rsvp: [2967] HELLO-MSG:  RSVP-HELLO: Received HELLO REQUEST
message from 10.1.1.2
2011 Mar 15 11:48:26.012371 rsvp: [2967] HELLO-MSG:  Received message with dst_address
(10.1.1.3) matching router ID
2011 Mar 15 11:48:26.012393 rsvp: [2967] HELLO-MSG:  RSVP-HELLO:
rsvp_hello_process_incoming_gr_message: restart_time 30000 recovery_time 120000
2011 Mar 15 11:48:26.012432 rsvp: [2967] HELLO-MSG:  Rcvd:Nbr 10.1.1.2  old_src_inst
845400891 new_src_inst 845400891, hc_event 0 hi_nbr_hello_state 1 hello_dst_inst
215306973, hi_my_src_inst 215306973
```

```
23:33:36: Outgoing Hello:
23:33:36: version:1 flags:0000 cksum:883C ttl:255 reserved:0 length:32
23:33:36: HELLO type HELLO REQUEST length 12:
23:33:36: Src_Instance: 0x6EDA8BD7, Dst_Instance: 0x00000000
23:33:36: RESTART_CAP type 1 length 12:
23:33:36: Restart_Time: 0x0000EA60, Recovery_Time: 0x0000EA60
```

Router 3 records this information into its database. Also, both neighbors maintain the neighbor status as UP. However, router 3's control plane fails at some point (for example, a primary RP failure). As a result, RSVP and TE lose their signaling information and states although data packets continue to be forwarded by the line cards.

When router 3 declares communication with router 2 lost, router 3 starts the restart time to wait for the duration advertised in router 2's restart time previously recorded (60 seconds). Routers 1 and 2 suppress all RSVP messages to router 3 except hellos. Router 3 keeps sending the RSVP PATH and RESV refresh messages to routers 4 and 5 so that they do not expire the state for the LSP; however, routers 1 and 3 suppress these messages for router 2.

When routers 1 and 3 receive the hello message from router 2, routers 1 and 3 check the recovery time value in the message. If the recovery time is 0, router 3 knows that router 2 was not able to preserve its forwarding information, and routers 1 and 3 delete the RSVP states that they had with router 2.

If the recovery time is greater than 0, router 1 sends router 2 PATH messages for each LSP that it had previously sent through router 2. If these messages were previously refreshed in summary messages, they are sent individually during the recovery time. Each of these PATH messages includes a Recovery_Label object that contains the label value received from router 2 before the failure.

When router 3 receives a PATH message from router 2, router 3 sends a RESV message upstream. However, router 3 suppresses the RESV message until it receives a PATH message. When router 2 receives the RESV message, it installs the RSVP state and reprograms the forwarding entry for the LSP.

Benefits of MPLS TE and RSVP Graceful Restart

State information recovery—RSVP graceful restart allows a node to do self recovery or to help its neighbor recover state information when there is an RP failure or the device has undergone an SSO.

Session information recovery—RSVP graceful restart allows session information recovery with minimal disruption to the network.

Increased availability of network services—A node can do a graceful restart to help itself or a neighbor recover its state by keeping the label bindings and state information, which provides a faster recovery of the failed node and does not affect currently forwarded traffic.

Configuring MPLS RSVP TE Graceful Restart

To alter the interval at which RSVP GR hellos are sent out by the local RSVP router, use the global **signalling hello graceful-restart refresh interval** command that applies to all neighbors. Graceful restart is on by default in Cisco NX-OS software and the default for the hello interval is 10 seconds.

```
switch(config-ip-rsvp)# signalling hello graceful-restart
switch(config-ip-rsvp)# signalling hello graceful-restart refresh interval
switch(config-ip-rsvp)# signalling hello graceful-restart refresh misses
switch(config-ip-rsvp)# signalling hello graceful-restart send restart-time
switch(config-ip-rsvp)# signalling hello graceful-restart send recovery-time
switch(config-ip-rsvp)# signalling hello reroute

switch(config-if)# ip rsvp signalling hello reroute
switch(config-if)# ip rsvp signalling hello reroute state-timeout refresh interval time
```

```
switch(config-if)# ip rsvp signalling hello reroute state-timeout refresh misses

switch(config-ip-rsvp)# signalling initial-retransmit-delay
switch(config-ip-rsvp)# signalling refresh interval
switch(config-ip-rsvp)# signalling refresh misses
switch(config-ip-rsvp)# signalling refresh reduction
switch(config-ip-rsvp)# signalling refresh reduction ack-delay
switch(config-ip-rsvp)# signalling refresh reduction bundle-max-size
switch(config-ip-rsvp)# signalling patherr state-removal
switch(config-ip-rsvp)# signalling rate-limit
```

To verify that this command has been configured correctly, use the **show ip rsvp graceful-restart** command.

RSVP Nonstop-Routing

RSVP nonstop routing (NSR) provides stateful high availability (HA) functionality to NX-OS. Two forms of service-level HA supported by RSVP NSR are as follows:

- Restartability—When an application crashes or hangs, it can be restarted by the system manager on the same supervisor.
- Switchover—If the kernel on the active supervisor fails, the active role can be switched to the standby supervisor.

Hello State Timer

The Hello State Timer (HST) provides for teardown of unprotected LSPs on non-FRR interfaces or LSPs with no backup on FRR interfaces by using RSVP hellos to detect a neighbor failure. The HST requires that the switch maintain hello communication with neighbors through which there are no protected LSPs running. If the hello communication with a neighbor is lost, HST initiates a PATH ERROR on the router that is upstream of the failed router and a PATH TEARDOWN on the neighbor that is downstream of the failed router (where there are instances running on both ends). The primary advantage of the HST is that it can detect an LSP failure and tear down the associated LSPs, and then free up requisite bandwidth more quickly than IGP.

Licensing Requirements for MPLS RSVP TE

Product	License Requirement
Cisco NX-OS	MPLS RSVP TE requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS RSVP TE

MPLS RSVP TE has the following prerequisites:

- Your network must support Multiprotocol Label Switching (MPLS).

Guidelines and Limitations for MPLS RSVP TE

MPLS RSVP TE has the following guidelines and limitations:

- The MPLS TE feature must be enabled.

Default Settings for MPLS RSVP TE

Table 12-1 lists the default settings for MPLS RSVP TE.

Table 12-1 Default Settings for MPLS RSVP TE

Parameters	Default
MPLS RSVP TE feature	Enabled

Configuring MPLS RSVP TE

This section includes the following topics:

Configuring RSVP Message Authentication

You can configure message authentication for MPLS RSVP TE.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **key chain** *key-chain-name*
3. **key** *key-identifier-number*
4. key-string
5. exit

6. **exit**
7. ip rsvp
8. **authentication** [**neighbor address** *IP-address*] **key-chain** *key-chain-name*
9. **authentication** [**neighbor address** *IP-address*] **type** {**md5** | **sha-1**}
10. **authentication** [**neighbor address** *IP-address*] **lifetime** *hh:mm:ss*
11. **authentication** [**neighbor address** *IP-address*] **window-size** *value*
12. **authentication** [**neighbor address** *IP-address*] **challenge**
13. exit
14. exit

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**key chain** ***key-chain-name*** **Example:** switch(config)# key chain ***key1*** switch(config-keychain)#	Enters the keychain management configuration mode and assigns a name for the key chain to be configured. The maximum size for the key-chain name is 63 alphanumeric characters.
Step 3	key *key-identifier-number* Example: switch(config-keychain)# key 4660 switch(config-keychain-key)#	Enters a keychain configuration ID and enters the keychain string configuration mode. The key-chain identifier number has a range from 0 to 65535.
Step 4	key-string Example: switch(config-keychain-key)# key-string qwertyui	Enters a keychain string for the keychain configuration ID.
Step 5	exit Example: switch(config-keychain-key)# exit switch(config-keychain)#	Exits the keychain string assignment mode and returns to keychain configuration ID mode.
Step 6	exit Example: switch(config-keychain)# exit switch(config)#	Exits the keychain configuration ID mode and returns to global configuration mode.
Step 7	ip rsvp Example: switch(config)# ip rsvp switch(config-ip-rsvp)#	Enters the RSVP configuration mode.
Step 8	**authentication** [**neighbor address** *IP-address*] **key-chain** *key-chain-name* Example: switch(config-ip-rsvp)# authentication neighbor address 10.0.0.2 key-chain key1	Activates RSVP cryptographic authentication for a neighbor or globally. The key-chain information is provided in a separate command. Use the **no** form of the command to disable global authentication.
Step 9	**authentication** [**neighbor address** *IP-address*] **type** {**md5** | **sha-1**} Example: switch(config-ip-rsvp)# authentication neighbor address 10.0.0.2 type sha-1	Specifies the algorithm used to generate cryptographic signatures messages for a neighbor or globally. By default, the authentication type is md5. To revert to the default md5 authentication configuration, use the **no** form of the command.

	Command	Purpose
Step 10	**authentication** [**neighbor address** *IP-address*] **lifetime** *hh:mm:ss* Example: switch(config-ip-rsvp)# authentication neighbor address 10.0.0.2 lifetime 10:30:30	Controls how long RSVP maintains security associations with a neighbor or globally. The default lifetime is 30 minutes. To revert to the default lifetime, use the **no** form of the command.
Step 11	**authentication** [**neighbor address** *IP-address*] **window-size** *value* Example: switch(config-ip-rsvp)# authentication neighbor address 10.0.0.2 window-size 2	Specifies the tolerance for out-of-sequence messages for a neighbor or globally. The default value is 1, which means all out-of-sequence messages are dropped. Use the **no** form of the command to return to the default configuration.
Step 12	**authentication** [**neighbor address** *IP-address*] **challenge** Example: switch(config-ip-rsvp)# authentication neighbor address 10.0.0.2 challenge	Makes RSVP use a challenge-response handshake with a neighbor.
Step 13	exit Example: switch(config-ip-rsvp)# exit switch(config)#	Exits RSVP configuration mode and returns to global configuration mode.
Step 14	exit Example: switch(config)# exit switch#	Exits global configuration mode.

Configuring Hello for MPLS RSVP TE

You can configure hellos for MPLS RSVP TE.

Note MPLS TE supports a single IGP process or instance. Do not configure MPLS TE in more than one IGP process or instance.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **ip rsvp**
3. **signalling hello graceful-restart**
4. **signalling hello graceful-restart refresh interval** *time*

5. **signalling hello graceful-restart refresh misses** *refresh-misses*
6. **signalling hello graceful-restart send restart-time** *time*
7. **signalling hello graceful-restart send recovery-time** *time*
8. **signalling hello reroute**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	ip rsvp Example: switch(config)# ip rsvp switch(config-ip-rsvp)#	Enters RSVP configuration mode.
Step 3	signalling hello graceful-restart Example: switch(config-ip-rsvp)# signalling hello graceful-restart	Globally enables signaling of node-based hellos for graceful restart functionality. The command is by default on at the global level. Use the **no** form of the command to disable sending of hellos for graceful restart.
Step 4	**signalling hello graceful-restart refresh interval** *time* Example: switch(config-ip-rsvp)# signalling hello graceful-restart refresh interval 15	Configures the interval at which RSVP graceful-restart hello messages are sent to each neighbor. The default value is 10 seconds. Use the **no** form of the command to return to the default behavior.
Step 5	**signalling hello graceful-restart refresh misses** *refresh-misses* Example: switch(config-ip-rsvp)# signalling hello graceful-restart refresh misses 6	Configures the number of consecutive missed hello message before a neighbor is declared down or unreachable. The default value is 4. Use the **no** form of the command to return to the default behavior.
Step 6	**signalling hello graceful-restart send restart-time** *time* Example: switch(config-ip-rsvp)# signalling hello graceful-restart send restart-time 20	Configures the restart time that is advertised in the Restart-Capability object in hello messages. The default restart time is 30 seconds. Use the **no** form of the command to return to the default behavior.
Step 7	**signalling hello graceful-restart send recovery-time** *time* Example: switch(config-ip-rsvp)# signalling hello graceful-restart send recovery-time 150	Configures the recovery time that is advertised in the Restart-Capability object in hello messages. The default recovery time is 120 seconds. Use the **no** form of the command to return to the default behavior.
Step 8	signalling hello reroute Example: switch(config-ip-rsvp)# signalling hello reroute	Globally enables the use of HST hellos. Sending of HST hellos is interface based, and configuration of reroute hello signaling is required at the global level and per-interface to enable sending of HST hellos on an interface. The command is by default off. Use the **no** form of the command to disable sending of reroute hellos.

Other Configurations for MPLS RSVP TE

You can use other configuration commands for MPLS RSVP TE.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **ip rsvp**
3. **signalling initial-retransmit-delay** *time*
4. **signalling refresh interval** *time*
5. **signalling refresh misses** *refresh-missed*
6. **signalling refresh reduction**
7. **signalling refresh reduction ack-delay** *time*
8. **signalling refresh reduction bundle-max-size** *value*
9. **signalling patherr state-removal**
10. **signalling rate-limit [burst** *value***] [period** *time***]**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	ip rsvp Example: switch(config)# ip rsvp switch(config-ip-rsvp)#	Enters RSVP configuration mode.
Step 3	**signalling initial-retransmit-delay** *time* Example: switch(config-ip-rsvp)# signalling initial-retransmit-delay 3	Configures the minimum amount of time that RSVP waits for an acknowledgement before retransmitting the same message. The default value is 1 second. Use the **no** form of the command to return to the default value.
Step 4	**signalling refresh interval** *time* Example: switch(config-ip-rsvp)# signalling refresh interval 40	Configures the frequency at which the RSVP state is refreshed. The default value is 30 seconds. Use the **no** form of the command to return to the default behavior.
Step 5	**signalling refresh misses** *refresh-missed* Example: switch(config-ip-rsvp)# signalling refresh misses 5	Specifies the number of refresh messages that can be missed before RSVP deems a state to be expired. The default value is 4. Use the **no** form of the command to return to the default behavior.
Step 6	signalling refresh reduction Example: switch(config-ip-rsvp)# signalling refresh reduction	Configures the RSVP refresh reduction. Use the **no** form of the command to disable RSVP refresh reduction. By default, refresh reduction is enabled.
Step 7	**signalling refresh reduction ack-delay** *time* Example: switch(config-ip-rsvp)# signalling refresh reduction ack-delay 300	Configures the maximum amount of time RSVP holds on to an acknowledgement before sending it. The default value is 250 ms (0.25sec). Use the **no** form of the command to return to the default value.
Step 8	**signalling refresh reduction bundle-max-size** *value* Example: switch(config-ip-rsvp)# signalling refresh reduction bundle-max-size 5000	Configures the bundle maximum send message size in the range from 0 to 65000 bytes, with the default being 4096 bytes. Set the value of 0 to disable sending of bundle messages.

	Command	Purpose
Step 9	`signalling patherr state-removal` `Example:` `switch(config-ip-rsvp)# signalling patherr state-removal`	Deletes the PATH state automatically when forwarding a PATH-ERROR message, which eliminates the need to send a subsequent PATH-TEAR message. Use the **no** form of the command to disable the path state deletion on a PATH ERROR.
Step 10	**signalling rate-limit** [**burst** *value*] [**period** *time*] `Example:` `switch(config-ip-rsvp)# signalling rate-limit burst 10 period 30`	Sets the rate limit for the number of messages that are sent to a neighboring router. The default burst is 8 messages in an interval of 20 ms. Use the **no** form of the command to return to the default behavior.

Verifying the MPLS RSVP TE Configuration

To display the MPLS RSVP TE configuration, perform one of the following tasks:

Command	Purpose
show ip rsvp	Displays global RSVP information.
show ip rsvp authentication	Displays the database for security associations that RSVP has established with neighbors. Use the **interface** or **neighbor** optional keyword to display the authentication information for the specified interface or neighbor. Use the **detail** optional keyword to display detailed authentication information.
show ip rsvp counters	Displays RSVP packet counters. Use the **interface** optional keyword to display interface RSVP packet counters. Use the **teardown** optional keyword to display RSVP teardown counters and the **all** optional keyword to display all counters.
show ip rsvp fast-reroute	Displays RSVP FRR information. This command is different than the Cisco IOS version, and is used to display RSVP centric FRR information. Use the **detail** optional keyword to display detailed fast-reroute information.
show ip rsvp graceful-restart	Displays restart information for RSVP.
show ip rsvp interface	Displays information about all interfaces with RSVP enabled. Use the **interface name** optional keyword to display information for the specified interface. Use the **detail** optional keyword to display detailed interface information and the **backup-tunnel** optional keyword to display backup-tunnel information known to RSVP.

Command	Purpose
show ip rsvp neighbor	Displays information about all RSVP neighbors. To display information about a neighbor, use the **neighbor** optional keyword. Use the **detail** optional keyword to display detailed neighbor information.
show ip rsvp reservation	Displays all reservations RSVP knows about on a router. Use the destination IP address, sender IP address, destination port, and/or source port information to filter the reservation information. Use the **detail** optional keyword to view detailed reservation display.
show ip rsvp sender	Displays all path-states RSVP knows about on a router. To display information about a particular sender or destination, use the sender IP address, destination IP address, destination port, and/or source port with the command. Use the **detail** optional keyword to view detailed sender information.
show ip rsvp session	Displays all sessions that RSVP knows about on a router. To display information about a particular destination, use the **destination IP-address** optional keyword with the command.
show ip rsvp hello instance	Displays RSVP hello instance information. Use the **detail** optional keyword to view detailed hello instance information.
show ip rsvp hello client lsp	Displays RSVP hello client LSP database. Use the **detail** optional keyword to view details of the LSP.
show ip rsvp hello client neighbor	Displays RSVP hello neighbor information. Use the **detail** optional keyword to view detailed hello neighbor information.
show ip rsvp signalling rate-limit	Displays RSVP globally configured signalling rate-limit information.
show ip rsvp signalling refresh reduction	Displays RSVP globally configured refresh reduction information.
show ip rsvp signalling refresh misses	Displays RSVP globally configured refresh misses information.
show ip rsvp signalling refresh interval	Displays RSVP globally configured refresh interval information.
show ip rsvp internal	Displays internal counters, event-history buffers, memory statistics or persistent store information. Use additional filters with these suboptions to filter the display the filtered information.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Verification Examples for MPLS RSVP TE

This section includes the following topics:

Example: Verifying the RSVP

The following example shows how to verify the global RSVP parameters:

```
switch# show ip rsvp
RSVP Process
        Supervisor State: Active
        Start Type: configuration [stateless]
        High Availability: Enabled [ok]
        Graceful Restart: disabled
        Hello State Timeout: (null:no-enum-table)
        Router id: 1.1.1.20
        Patherr State Removal: Disabled
        Local Epoch: 0xab446c

Registered RSVP Clients
        MPLS TE [Service-Access-Point 288, ID 1, Batch-Time 50 msec]
                [Listener: Flags 0x7, Events 0x7ffff, ClientHintLen 24]

Message Bundling
        Enabled [Transmit-delay 50 msec, Max-Size 4096 bytes]

Refresh Parameters
        Interval 45 sec, Miss-Limit 4

Refresh-Reduction
```

```
        Enabled [Initial-Retransmit-Delay 5000 msec]
        [Rapid-Retransmit Disabled, Ack-Delay 400 msec]

Rate-Limit
        Disabled [Limit 100 messages, Interval 1000 msec]

GR Recovery Timer
        Not running

Authentication
        Disabled
```

Example: Verifying the RSVP Neighbor

The following example shows how to verify the RSVP neighbor:

```
switch# show ip rsvp neighbor
Address         Interface      RouterID        State    Expiry     LastSend
3.0.206.6       Ethernet1/7    1.1.1.6         UP,RR    14 minutes    4 sec
```

Example: Verifying the RSVP Reservation

The following example shows how to verify RSVP reservations:

```
switch# show ip rsvp reservation

Total Reservation States: 3000
To            From          Pro DPort Sport Next Hop      I/F           Fi
1.1.1.13      1.1.1.20      0   20000 19    3.0.206.6     Eth1/7        SE
1.1.1.13      1.1.1.20      0   20001 19    3.0.206.6     Eth1/7        SE
.
.
.
1.1.1.13      1.1.1.20      0   22997 17    3.0.206.6     Eth1/7        SE
1.1.1.13      1.1.1.20      0   22998 17    3.0.206.6     Eth1/7        SE
1.1.1.13      1.1.1.20      0   22999 17    3.0.206.6     Eth1/7        SE
```

Example: Verifying the RSVP Sender

The following example shows how to verify the RSVP senders:

```
switch# show ip rsvp sender
Total Sender States: 3000
To              From            Pro DPort Sport Prev Hop        I/F
1.1.1.13        1.1.1.20        0   20000 19    none            None
1.1.1.13        1.1.1.20        0   20001 19    none            None
.
.
.
1.1.1.13        1.1.1.20        0   22998 17    none            None
1.1.1.13        1.1.1.20        0   22999 17    none            None
```

Example: Verifying the RSVP Sessions

The following example shows how to verify RSVP sessions:

```
switch(config-if-te)# show ip rsvp session
Total Sessions: 4
Type Destination     DPort Proto/ExtTunID  PSBs  RSBs  Reqs  PXSBs RXSBs
LSP4 10.10.10.10     10    10.10.10.15     1     1     0     1     0
LSP4 10.10.10.10     11    10.10.10.15     1     1     0     1     0
LSP4 10.10.10.10     12    10.10.10.15     1     1     0     1     0
LSP4 10.10.10.10     13    10.10.10.15     1     1     0     1     0
```

Example: Verifying the RSVP Signaling Rate Limit

The following example shows how to verify the RSVP signaling rate limit:

```
switch# show ip rsvp signalling rate-limit
Rate-Limiting: Disabled
  Limit: 100
  Interval (msec): 1000
switch# show ip rsvp signalling refresh ?
  interval   Display interval for refresh messages
  misses     Display misses required to trigger state timeout
  reduction  Display refresh reduction parameters
```

Example: Verifying the RSVP Signaling Refresh Interval

The following example shows how to verify the RSVP signaling refresh interval:

```
switch# show ip rsvp signalling refresh interval
Refresh interval (sec): 45
```

Example: Verifying the RSVP Signaling Refresh Misses

The following example shows how to verify the RSVP signaling refresh misses:

```
switch# show ip rsvp signalling refresh misses
Refresh misses: 4
```

Example: Verifying the RSVP Signaling Refresh Reduction

The following example shows how to verify the RSVP signaling refresh reduction:

```
switch# show ip rsvp signalling refresh reduction
Refresh Reduction: Enabled
  ACK delay (msec): 400
  Initial retransmit delay (msec): 5000
  Local epoch: 0xab446c
  Message IDs: in use 6000, total allocated 33005, freed 27005
```

Example: Verifying the RSVP Counters

The following example shows how to verify the RSVP counters:

```
switch# show ip rsvp counters
All Interfaces    Recv       Xmit                          Recv       Xmit
Packet            40641      40847     PacketError         0          0
Path              0          301       Resv                0          0
```

```
PathError          0          0          ResvError          0          0
PathTear           0          0          ResvTear           0          0
ResvConf           0          0          RTearConf          0          0
Ack                5          2          SRefresh           38341      38510
Hello              0          0          IntegrityChallenge 0          0
IntegrityResponse  0          0

Bundle             1672       1875
Path               0          9192       Resv               9135       0
PathError          3559       0          ResvError          0          0
PathTear           0          6000       ResvTear           3557       0
Ack                0          0
switch#
```

Example: Verifying All of the RSVP Counters

The following example shows how to verify all of the RSVP counters:

```
switch# show ip rsvp counters all
Teardown Reason                 Path       Resv
UNSPECIFIED                     0          0
PATH TIMEOUT                    0          0
RESV TIMEOUT                    0          0
SIGNALED                        4448       4448
MGMT                            0          0
POLICY                          0          0
PROXY                           3069       3069
NO_RESOURCES                    0          0
PREEMPTED                       0          0
MSG_ERROR                       0          0
INTERNAL                        0          0
TRAFFIC_CONTROL                 0          0
POLICY_SYNC                     0          0
GR_TIMEOUT                      0          0
LINK_NBOR_DOWN                  0          0
LOCAL-SEND_PERR_PSR             0          0
NETWORK_PERR_PSR                0          0
HST_TIMEOUT                     0          0
PLR BACKUP DELETE               0          0
CLI-CLEAR                       0          0
RESTART-COMMAND                 0          0
INTERFACE-DELETE                0          0
Sent:
  Messages successfully authenticated:    0
  Messages internal failure:              0
Received:
  Messages successfully authenticated:    0
  Total receive errors:                   0
  Receive Errors:
    Missing INTEGRITY object:             0
    Incorrect digest:                     0
    Digest type mismatch:                 0
    Duplicate sequence number:            0
    Out-of-range sequence number:         0
Challenge Handshake:
  Challenges Received:                    0
  Challenges Responded:                   0
  Initiations:                            0
  Timeouts:                               0
  Retransmissions:                        0
  Responses Received:                     0
  Drops during Challenge:                 0
```

```
  Incorrect challenge response:           0
  Duplicate challenge response:           0
  Late challenge response:                0
All Interfaces    Recv       Xmit                          Recv       Xmit
Packet            40645      40851     PacketError         0          0
Path              0          301       Resv                0          0
PathError         0          0         ResvError           0          0
PathTear          0          0         ResvTear            0          0
ResvConf          0          0         RTearConf           0          0
Ack               5          2         SRefresh            38345      38514
Hello             0          0         IntegrityChallenge  0          0
IntegrityResponse 0          0

Bundle            1672       1875
Path              0          9192      Resv                9135       0
PathError         3559       0         ResvError           0          0
PathTear          0          6000      ResvTear            3557       0
Ack               0          0
```

Example: Verifying the RSVP Counters for Teardown

The following example shows how to verify the RSVP counters for teardown:

```
switch# show ip rsvp counters teardown
Teardown Reason                Path       Resv
UNSPECIFIED                    0          0
PATH TIMEOUT                   0          0
RESV TIMEOUT                   0          0
SIGNALED                       4448       4448
MGMT                           0          0
POLICY                         0          0
PROXY                          3069       3069
NO_RESOURCES                   0          0
PREEMPTED                      0          0
MSG_ERROR                      0          0
INTERNAL                       0          0
TRAFFIC_CONTROL                0          0
POLICY_SYNC                    0          0
GR_TIMEOUT                     0          0
LINK_NBOR_DOWN                 0          0
LOCAL-SEND_PERR_PSR            0          0
NETWORK_PERR_PSR               0          0
HST_TIMEOUT                    0          0
PLR BACKUP DELETE              0          0
CLI-CLEAR                      0          0
RESTART-COMMAND                0          0
INTERFACE-DELETE               0          0
```

Example: Verifying the RSVP Counters Authentication

The following example shows how to verify the RSVP counters authentication:

```
switch# show ip rsvp counters authentication
Sent:
  Messages successfully authenticated:    0
  Messages internal failure:              0
Received:
  Messages successfully authenticated:    0
  Total receive errors:                   0
```

```
  Receive Errors:
    Missing INTEGRITY object:             0
    Incorrect digest:                     0
    Digest type mismatch:                 0
    Duplicate sequence number:            0
    Out-of-range sequence number:         0
Challenge Handshake:
  Challenges Received:                    0
  Challenges Responded:                   0
  Initiations:                            0
  Timeouts:                               0
  Retransmissions:                        0
  Responses Received:                     0
  Drops during Challenge:                 0
  Incorrect challenge response:           0
  Duplicate challenge response:           0
  Late challenge response:                0
```

Example: Verifying the RSVP FRR

The following example shows how to verify the RSVP FRR:

```
switch# show ip rsvp fast-reroute
    A - Active      R - Ready      U - Unassigned
Destination     TunID Source          Backup         Protected-I/f   Hop

State
Fast-Reroute Summary:
  Total Reroutable Paths: 0
    Active: 0, Ready: 0, Unassigned: 0
```

Example: Verifying the RSVP Hello Client LSP

The following example shows how to verify the RSVP hello client LSP:

```
switch# show ip rsvp hello client lsp

Local           Remote          tun_id lsp_id subgrp_orig     subgrp_id FLAGS
```

Example: Verifying the RSVP Hello Graceful-Restart

The following example shows how to verify the RSVP hello graceful-restart:

```
switch(config-if-te)# show ip rsvp hello graceful-restart
Graceful Restart: Enabled (full mode)
  Refresh interval: 10000 msecs
  Refresh misses: 4
  DSCP: 0xc0
  Advertised restart time: 30000 msecs
  Advertised recovery time: 120000 msecs
  Maximum wait for recovery: 3600000 msecs
```

Example: Verifying the RSVP Hello Instance

The following example shows how to verify the RSVP hello instance:

```
switch# show ip rsvp hello instance

Active Instances:
  - None -

Passive Instances:
  - None -
```

Example: Verifying the RSVP Interface

The following example shows how to verify the RSVP interface:

```
switch(config-if-te)# show ip rsvp interface
Interface           Ifindex    IOD    MPLS     Config   State
Ethernet2/2         0x1a081000 37     enabled  None     Up
Ethernet2/7         0x1a086000 42     enabled  None     Down
loopback0           0x14000000 45     enabled  None     Up
```

Additional References for MPLS RSVP TE

For additional information related to implementing MPLS RSVP TE, see the following sections:

- Related Document, page 12-200
- MIBs, page 12-200

Related Document

Related Topic	Document Title
MPLS TE commands	*Cisco NX-OS MPLS Command Reference*

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download Cisco MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS RSVP TE

Table 12-2 lists the release history for this feature.

Table 12-2 Feature History for MPLS RSVP TE

Feature Name	Releases	Feature Information
MPLS RSVP TE	5.2(1)	This feature was introduced.

Configuring the Path Selection Metric for MPLS TE Tunnels

This chapter describes how to configure the path selection metric for Multiprotocol Label Switching (MPLS) traffic engineering (TE) tunnels on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About the Path Selection Metric for MPLS TE Tunnels

Certain tunnels carry voice traffic, which requires low delay, and other tunnels carry data.You can configure the path selection metric for TE tunnels on a global or per-tunnel basis. You can specify the TE link metric on an interface, or let it default to the IGP link metric. You can also specify the link metric for path selection for these low-delay traffic tunnels and let the other tunnels use the Interior Gateway Protocol (IGP) metric for path selection. MPLS TE supports Intermediate System-to-Intermediate System (IS-IS) and Open Shortest Path First (OSPF) as the IGP.

IGP floods two metrics for every MPLS TE link as follows:

- IGP link metric, which is the IGP determined metric for the link.
- Path selection metric, which is the TE link metric for the link. This metric defaults to the IGP link metric, but you can specify the TE link metric by using the **mpls traffic-eng administrative-weight** command on the interface.

Licensing Requirements for the Path Selection Metric for MPLS TE Tunnels

Product	License Requirement
Cisco NX-OS	Configurable path selection metric for tunnels requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for the Path Selection Metric for MPLS TE Tunnels

The path selection metric for MPLS TE tunnels has the following prerequisites:

- You must enable MPLS TE on your device. See the "Configuring MPLS TE" section on page 10-139.

Guidelines and Limitations for the Path Selection Metric for MPLS TE Tunnels

The path selection metric has the following configuration guidelines and limitations:

- Unless explicitly configured, the TE link metric (administrative-weight) for a given link is the IGP link metric. When the TE link metric is used to represent a link property that is different from the cost/distance, you must configure every network link that can be used for TE tunnels with a TE link metric that represents that property. Failure to do so might cause tunnels to use unexpected paths.
- You cannot configure MPLS TE over the logical generic routing encapsulation (GRE) tunnel interface.

- MPLS TE supports only a single IGP process or instance. Multiple IGP processes or instances are not supported and you should not configure MPLS TE in more than one IGP process or instance. You might configure MPLS TE in multiple OSPF areas or both IS-IS levels.

Default Settings for the Path Selection Metric for MPLS TE Tunnels

Table 13-1 lists the default settings for the path selection metric for MPLS TE tunnels.

Table 13-1 Default Settings for the Path Selection Metric for MPLS TE Tunnels

Parameters	Default
Global path selection metric type	TE
Interface path selection metric type	IGP
TE tunnel interface path selection metric type	TE
TE link metric/administrative-weight	IGP link metric

Note Although both the global path selection metric and TE tunnel path selection metric default to TE, because the interface TE link metric defaults to the IGP link metric, the effective default configuration uses the IGP link metrics.

Configuring the Path Selection Metric for MPLS TE Tunnels

This section includes the following topics:

Configuring the Global Path Selection Metric Type for MPLS TE Tunnels

You can configure the path selection to use either the IGP metric or the TE metric for all MPLS TE tunnels.

Note The configured TE tunnel interface path selection metric type takes precedence over the global path selection metric type.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng configuration**
3. **path-selection metric** {**igp** | **te**}
4. (Optional) **show mpls traffic-eng tunnels tunnel-te** *number*
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**mpls traffic-eng configuration** **Example:** switch(config)# mpls traffic-eng configuration switch(config-te)#	Enters MPLS TE configuration mode.
Step 3	**path-selection metric** {**igp** | **te**} **Example:** switch(config-te)# path-selection metric te	Specifies the metric type to use if a metric type is not explicitly configured for an MPLS TE tunnel. If you configure the TE path selection metric type, you can also configure the MPLS TE administrative weight on each TE tunnel. The default is IGP.
Step 4	**show mpls traffic-eng tunnels tunnel-te** *number* **Example:** switch(config-if-te)# show mpls traffic-eng tunnels tunnel-te 0	(Optional) Displays information about the MPLS TE tunnel configuration.
Step 5	**copy running-config startup-config** **Example:** switch(config-if-te)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring the Path Selection Metric Type for a TE Tunnel

You can override the global tunnel path selection metric on a per-TE tunnel interface, and you can also override the IGP link metric on any link by configuring the TE administrative weight on that link.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **path-selection metric** {**igp** | **te**}
4. (Optional) **interface** type/*number*

5. (Optional) **mpls traffic-eng administrative-weight** *weight*
6. (Optional) **show mpls traffic-eng tunnels tunnel-te** *number*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1` `switch(config-if-te)#`	Enters TE interface configuration mode. The *number* range is from 0 to 65503.
Step 3	**path-selection metric** {**igp** | **te**} **Example:** `switch(config-if-te)# path-selection metric te`	Specifies the metric type to use for the MPLS TE tunnel path selection. The default is TE.
Step 4	**interface** *type/number* **Example:** `switch(config-if-te)# interface ethernet 2/1` `switch(config-if)#`	(Optional) Enters interface configuration mode. Use **?** to see a list of supported interfaces.
Step 5	**mpls traffic-eng administrative-weight** *weight* **Example:** `switch(config-if)# mpls traffic-eng administrative weight 20`	(Optional) Overrides the IGP administrative weight (cost) of the link. Configure this parameter if you configured the path selection metric type as TE and you want to specify the TE link cost/administrative-weight rather than defaulting to the IGP cost.
Step 6	**show mpls traffic-eng tunnels tunnel-te** *number* **Example:** `switch(config-if)# show mpls traffic-eng tunnels tunnel-te 0`	(Optional) Displays information about the MPLS TE tunnel configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the Path Selection Metric Configuration for MPLS TE Tunnels

To display the path selection metric for MPLS TE tunnels, perform one of the following tasks:

Command	Purpose
show mpls traffic-eng topology	Displays the TE and IGP metrics for each link.

Command	Purpose
show mpls traffic-eng tunnels	Displays the link metric used for tunnel path selection.
show mpls traffic-eng link-management summary	Displays each link's administrative weight and whether it was inherited from the IGP or is an override with the TE administrative weight.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

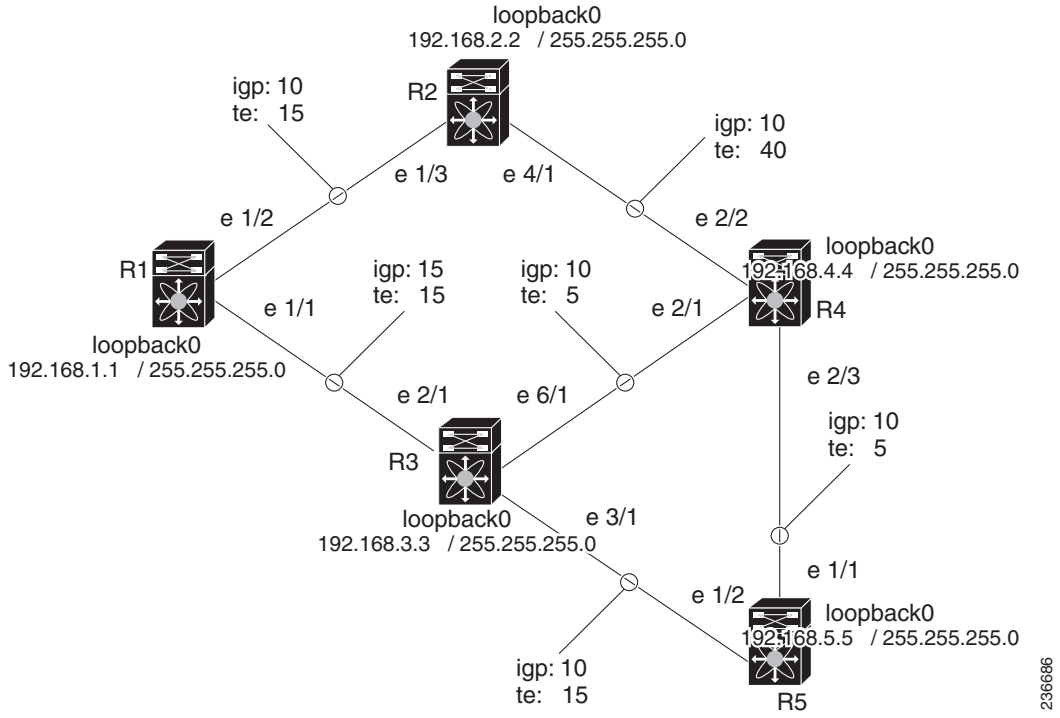
Configuration Examples for the Path Selection Metric for MPLS TE Tunnels

The section shows how to configure the link metric type that you can use for the tunnel path selection and how to configure the link metrics. The configuration commands included in this section allow you to specify the metric type for path selection and assign metrics to links. Additional commands are required to fully configure the example scenario. For example, you need the IGP commands to continue TE and the link interface commands to enable TE and specify the available bandwidth.

The example in this section supports the simple network technology shown in Figure 13-1.

Note This example applies only to dynamic tunnels, not to ones with an explicit path.

Figure 13-1 Network Topology



In Figure 13-1, the topology shows the following:

- Tunnel1 and Tunnel2 connects from R1 (headend) to R4 (tailend).
- Tunnel3 connects from R1 to R5.
- The path selection for Tunnel1 and Tunnel3 should use a metric that represents a link delay because these tunnels carry voice traffic.
- The path selection for Tunnel2 should use IGP metrics because MPLS TE carries data traffic with no delay requirement.

The following configuration fragments for each of the routers show the configuration that relates to link metrics and their use in the tunnel path selection. TE metrics that represent a link delay must be configured for the network links on each of the routers, and the three tunnels must be configured on R1.

These configuration fragments force Tunnel1 to take path R1-R3-R4, Tunnel2 to take path R1-R2-R4, and Tunnel3 to take path R1-R3-R4-R5 (if the links have sufficient bandwidth to accommodate the tunnels).

R1 Configuration

```
interface eth1/1
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
interface eth1/2
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric

interface Tunnel-te 1                           !Tunnel1 uses TE metric (default)
                                                !for path selection
 ip unnumbered loopback0
 destination 192.168.4.4
 bandwidth 1000
 path-option 1 dynamic
```

```
interface Tunnel-te 2                              !Tunnel2 uses IGP metric
                                                   !for path selection

 ip unnumbered loopback0
 destination 192.168.4.4
 bandwidth 1000
 path-option 1 dynamic
 path-selection metric igp                          !Use IGP cost for path selection.

interface Tunnel 3                                  !Tunnel3 uses TE metric (default)
                                                    !for path selection
 ip unnumbered loopback0
 destination 192.168.5.5
 bandwidth 1000
 path-option 1 dynamic
```

R2 Configuration

```
interface eth1/3
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
 interface eth4/1
mpls traffic-eng administrative-weight 40       !TE metric different from IGP metric
```

R3 Configuration

```
interface eth2/1
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
interface eth3/1
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
interface eth6/1
 mpls traffic-eng administrative-weight 5        !TE metric different from IGP metric
```

R4 Configuration

```
interface eth2/2
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
interface eth2/1
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
interface eth2/3
 mpls traffic-eng administrative-weight 5        !TE metric different from IGP metric
```

R5 Configuration

```
interface eth1/2
 mpls traffic-eng administrative-weight 15       !TE metric different from IGP metric
interface eth1/1
 mpls traffic-eng administrative-weight 5        !TE metric different from IGP metric
```

Additional References for MPLS TE Tunnels

The following sections provide references related to the path selection metric for MPLS TE tunnels.

Related Document

Related Topic	Document Title
MPLS TE commands	*Cisco NX-OS Multiprotocol Label Switching Command Reference*

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download Cisco MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for the Path Selection Metric for MPLS TE Tunnels

Table 13-2 lists the release history for this feature.

Table 13-2 Feature History for the Path Selection Metric for MPLS TE Tunnels

Feature Name	Releases	Feature Information
Path selection metric	5.2(1)	This feature was introduced.

Configuring LSP Attributes for MPLS TE

This chapter describes how to configure label switched path (LSP) attributes for path options that are associated with Multiprotocol Label Switching (MPLS) traffic engineering (TE) tunnels on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About LSP Attributes for MPLS TE

You can configure an LSP attribute list and associate it with one or more MPLS TE tunnels. This LSP attribute list supports multiple LSP attributes such as bandwidth and priority. You can also configure a TE tunnel to fall back temporarily to use path options that can reduce bandwidth constraints.

A port-profile can be used to provide a template of configuration for an entire tunnel. The attribute list can be used to apply a template for the configuration for one or more path options.

This section includes the following topics:

LSP Attribute Lists

Cisco NX-OS tunneling interfaces have many parameters that are associated with MPLS TE. Typically, you configure these parameters on an interface. Many of these attributes determine tunnel-specific properties, such as load sharing for the tunnel. These parameters are unrelated to the particular LSP in use by the tunnel. However, some of the tunneling parameters apply to the LSP that the tunnel uses.

You can configure the LSP-specific properties using an LSP attribute list. An LSP attribute list can contain values for each LSP-specific parameter that is configurable for a TE tunnel. You can specify the following LSP attributes in an attribute list:

- Attribute flags for links that make up the LSP
- Automatic bandwidth configuration
- LSP bandwidth from the global pool
- Disabling reoptimization of the LSP
- LSP priority
- Protection failure
- Recording the route used by the LSP

You can relist all attributes or remove specific attributes from an LSP attribute list.

Based on your requirements, you can configure LSP attribute lists with different sets of attributes for different path options. LSP attribute lists also provide an easy way to configure multiple TE tunnels to use the same LSP attributes. You can reference the same LSP attribute list to configure LSP-specific parameters for one or more TE tunnels.

Autobandwidth

If you configure TE automatic bandwidth (autobandwidth) adjustment for a tunnel, traffic engineering automatically adjusts the bandwidth allocation for the TE tunnel based on its measured usage of the bandwidth of the tunnel.

TE autobandwidth samples the average output rate for each tunnel that is marked for automatic bandwidth adjustment. For each marked tunnel, TE periodically adjusts the allocated bandwidth for the tunnel to be the largest sample for the tunnel since the last adjustment. The default reoptimization setting for TE autobandwidth is every 24 hours.

You can configure the frequency with which the tunnel bandwidth is adjusted and the allowable range of adjustments per tunnel. You can also configure the sampling interval and the interval over which to average the tunnel traffic to obtain the average output rate per tunnel.

For more information on automatic bandwidth adjustment for TE tunnels, see the "MPLS Traffic Engineering (TE)—Automatic Bandwidth Adjustment for TE Tunnels" chapter.

You can also select a path option to override the bandwidth that is configured on a TE tunnel. This feature enables you to override the bandwidth that is configured or recalculated by the automatic bandwidth adjustment if you have enabled the path option with the bandwidth override. See the "Path Option Selection with Bandwidth Override" section on page 14-215.

Note You cannot configure both the bandwidth in the LSP attribute list and the bandwidth override because they are mutually exclusive.

Path Option Selection for MPLS TE Tunnel LSPs

This section includes the following topics:

Constraint-Based Routing and Path Option Selection

MPLS TE automatically establishes and maintains LSPs across the backbone by using the Resource Reservation Protocol (RSVP). The path that an LSP uses is determined by the LSP resource requirements and network resources, such as bandwidth. TE tunnels are calculated at the LSP head based on a fit between required and available resources (constraint-based routing).

A TE tunnel establishes an LSP based on dynamic or explicit path options in order of preference. However, the bandwidth and other attributes configured on the TE tunnel allow the setup of an LSP only if LSP path options satisfy the constraints. If a path cannot be found that satisfies the configured path options, then the tunnel is not set up.

You can configure the path option for bandwidth override as a fallback path option that allows overriding the bandwidth configured on the TE tunnel interface. For example, you can configure a path option that sets the bandwidth to zero (0), which effectively removes the bandwidth constraint imposed by the constraint-based routing calculation.

If the bandwidth is the only LSP attribute that you need to set on the path option, use the path option for the bandwidth override, which is the simplest way to configure multiple path options with decreasing bandwidth constraints.

Note You cannot configure both the bandwidth in the LSP attribute list and the bandwidth override because they are mutually exclusive.

Tunnel Reoptimization and Path Option Selection

Reoptimization occurs when a device with TE tunnels periodically examines tunnels with established LSPs to learn if better LSPs are available. If a better LSP is available, the device signals the better LSP. If the signaling is successful, the device replaces the older LSP with the new, better LSP.

You can trigger reoptimization using a configurable timer, a reoptimize command, or a configuration change that requires the resignaling of a tunnel. Autobandwidth, for example, uses a timer to set the frequency of reoptimization based on the bandwidth path option attribute. The path option for bandwidth override can switch between the bandwidth that is configured on the TE tunnel interface and the bandwidth that is configured on a specific path option. This override increases the success of signaling an LSP for the TE tunnel.

When you configure the bandwidth override on a path option, TE reoptimizes the bandwidth every 30 seconds to reestablish the bandwidth that is configured on the tunnel (see the "Configuring a Path Option for Bandwidth Override" section on page 14-224).

You can disable reoptimization of an LSP in an LSP attribute list and apply this LSP attribute list to a path option.

Path Option Selection with Bandwidth Override

When you enable the bandwidth override path option, you can configure bandwidth parameters on a specific path option. When an LSP is signaled using a path option with a configured bandwidth, the bandwidth that is associated with the path option is signaled instead of the bandwidth that is configured directly on the tunnel.

You can configure multiple path options that reduce the bandwidth constraint each time the headend of a tunnel fails to establish an LSP.

The following example shows three **path-option** commands:

```
bandwidth 1000
path-option 1 explicit name path1
path-option 2 explicit name path2 bandwidth 500
path-option 3 dynamic bandwidth 0
```

The device selects a path option for an LSP in order of preference, as follows:

- The device signals an LSP using path options that start with path option 1.

  The device signals an LSP with the 1000-kbps bandwidth configured on the tunnel interface because path option 1 has no bandwidth configured.

- If 1000 kbps is not available, the device tries to establish an LSP using path option 2.

  Path option 2 has a 500-kbps bandwidth configured, which reduces the bandwidth constraint from the original 1000-kbps configured on the tunnel interface.

- If 500 kbps is not available, the device tries to establish an LSP using path option 3.

  Path option 3 is configured as dynamic and has a bandwidth of 0. The device establishes the LSP if an IP path exists to the destination and all other tunnel constraints are met.

Licensing Requirements for LSP Attributes for MPLS TE

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	The MPLS TE LSP attributes feature requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for LSP Attributes for MPLS TE

LSP attributes for MPLS TE tunnels have the following prerequisite:

- You must enable the MPLS TE feature. See the "Configuring MPLS TE" section on page 10-139.

Guidelines and Limitations for LSP Attributes for MPLS TE

LSP attributes for MPLS TE tunnels have the following configuration guidelines and limitations:

- You must explicitly configure the LSP attributes.

Default Settings for LSP Attributes for MPLS TE

Table 14-1 lists the default settings for LSP attributes.

Table 14-1 Default Settings for LSP Attributes

Parameters	Default
Affinity	0, mask 0
Auto bandwidth	disabled
Bandwidth	0
Priority	7 7
Protection fast reroute	Disabled
Record route	Disabled

Configuring LSP Attributes for MPLS TE

This section includes the following topics:

Configuring LSP Attributes in an MPLS TE Tunnel

You can configure LSP attributes in an MPLS TE tunnel. These values are overridden by an LSP attribute list that is associated with this MPLS TE tunnel.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **destination** {*ip-address*}
4. (Optional) **autoroute announce**
5. (Optional) **bandwidth** *kbps*
6. (Optional) **priority** *setup-priority* [*hold-priority*]
7. (Optional) **show interface tunnel-te** *number*
8. (Optional) **show mpls traffic-eng tunnels tunnel-te** *number*
9. (Optional) **show running config interface tunnel-te** *number*
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**destination** {*ip-address*} **Example:** `switch(config-if-te)# destination 10.10.10.12`	Specifies the destination of the tunnel for this path option. The *ip-address* argument is the IP address of the host destination expressed in decimal in four-part, dotted notation.
Step 4	**autoroute announce** **Example:** `switch(config-if-te)# autoroute announce`	(Optional) Specifies that the IGP should use the tunnel (if the tunnel is up) in its enhanced shortest path first (SPF) calculation.
Step 5	**bandwidth** *kbps* **Example:** `switch(config-if-te)# bandwidth 1000`	(Optional) Configures the bandwidth required for an MPLS TE tunnel and assigns it to the global pool. The *kbps* argument is the bandwidth, in kilobits per second, set aside for the MPLS TE tunnel. The range is from 1 to 4294967295.
Step 6	**priority** *setup-priority* [*hold-priority*] **Example:** `switch(config-if-te)# priority 1 1`	(Optional) Sets the priority to be used when the system determines which existing tunnels are eligible to be preempted. The *setup-priority* argument is the priority used when signaling an LSP for this tunnel to determine which existing tunnels can be preempted. Valid values are from 0 to 7. A lower number indicates a higher priority. An LSP with a setup priority of 0 can preempt any LSP with a nonzero priority. The *hold-priority* argument is the priority associated with an LSP for this tunnel to determine if it should be preempted by other LSPs that are being signaled. Valid values are from 0 to 7, where a lower number indicates a higher priority.
Step 7	**show interface tunnel-te** *number* **Example:** `switch(config-if-te)# show interface tunnel-te 1`	(Optional) Displays information about the TE tunnel.
Step 8	**show mpls traffic-eng tunnels tunnel-te** *number*	(Optional) Displays the MPLS TE tunnels for the configured tunnel *number*.

	Command	Purpose
Step 9	**show running config interface tunnel-te** *number*	(Optional) Displays the running configuration of the interface MPLS TE tunnels for the configured tunnel *number*.
Step 10	**copy running-config startup-config** **Example:** switch(config-if-te)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring an LSP Attribute List

You can configure an LSP attribute list with the desired attributes to apply on a path option. You can also add or modify an attribute in an existing LSP attribute list, or use the **no** subcommand to remove an attribute from an existing attribute list.

Note You cannot configure both the bandwidth in the LSP attribute list and the path-option bandwidth override because they are mutually exclusive.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng configuration**
3. **lsp attributes** *name*
4. (Optional) **affinity** *value* [**mask** *value*]
5. (Optional) **auto-bw** [**frequency** *secs*] [**max-bw** *kbps*] [**min-bw** *kbps*] [**collect-bw**]
6. (Optional) **bandwidth** *kbps*
7. (Optional) **list**
8. (Optional) **lockdown**
9. (Optional) **priority** *setup-priority* [*hold-priority*]
10. (Optional) **protection fast-reroute**
11. (Optional) **record-route**
12. (Optional) **no** *sub-command*
13. (Optional) **show mpls traffic-eng lsp attributes** [*name*]
14. (Optional) **show running mpls traffic-eng**
15. (Optional) **show mpls traffic-eng tunnels tunnel-te** *number*
16. (Optional) **show running config interface tunnel-te** *number*
17. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls traffic-eng configuration** **Example:** `switch(config)# mpls traffic-eng configuration` `switch(config-te)#`	Enters MPLS TE configuration mode.
Step 3	**lsp attributes** *string* **Example:** `switch(config-te)# lsp attributes 1` `switch(config-lsp-attr)#`	Configures an LSP attribute list and enters LSP attributes configuration mode. The *name* argument identifies a specific LSP attribute list and can be any case-sensitive, alphanumeric string up to 63 characters.
Step 4	**affinity** *value* [**mask** *value*] **Example:** `switch(config-lsp-attr)# affinity 0 mask 0`	(Optional) Specifies attribute flags for links that comprise an LSP. The *value* argument is a value required for links that make up an LSP. Values of the bits are either 0 or 1. The **mask** *value* keyword argument combination indicates which attribute values should be checked. If a bit in the mask is 0, an attribute value of the link or that bit is irrelevant. If a bit in the mask is 1, the attribute value of that link and the required affinity of the LSP for that bit must match.
Step 5	**auto-bw** [**frequency** *secs*] [**max-bw** *kbps*] [**min-bw** *kbps*] [**collect-bw**] **Example:** `switch(config-lsp-attr)# auto-bw`	(Optional) Specifies automatic bandwidth configuration with the following options: • The **frequency** *secs* keyword argument combination specifies the interval between bandwidth adjustments. The range is from 300 to 604800 seconds. • The **max-bw** *kbps* keyword argument combination specifies the maximum automatic bandwidth, in kbps, for this path option. The range is from 1 to 4294967295. • The **min-bw** *kbps* keyword argument combination specifies the minimum automatic bandwidth, in kbps, for this path option. The range is from 1 to 4294967295. • The **collect-bw** keyword collects output rate information for the path option but does not adjust the bandwidth of the path option.
Step 6	**bandwidth** *kbps* **Example:** `switch(config-lsp-attr)# bandwidth 5000`	(Optional) Specifies the LSP bandwidth. The *kbps* argument is the number of kilobits per second set aside for the path option. The range is from 1 to 4294967295.

	Command	Purpose
Step 7	**list** **Example:** `switch(config-lsp-attr)# list`	(Optional) Displays the contents of the LSP attribute list.
Step 8	**lockdown** **Example:** `switch(config-lsp-attr)# lockdown`	(Optional) Disables reoptimization of the LSP.
Step 9	**priority** *setup-priority* [*hold-priority*] **Example:** `switch(config-lsp-attr)# priority 1 1`	(Optional) Specifies the LSP priority. The *setup-priority* argument is used when signaling an LSP to determine which existing LSPs can be preempted. The range is from 0 to 7, where a lower number indicates a higher priority. Therefore, an LSP with a setup priority of 0 can preempt any LSP with a nonzero priority. The *hold-priority* argument is associated with an LSP to determine if it should be preempted by other LSPs that are being signaled. The range is from 0 to 7, where a lower number indicates a higher priority.
Step 10	**protection fast-reroute** **Example:** `switch(config-lsp-attr)# protection fast-reroute`	(Optional) Enables failure protection on the LSP.
Step 11	**record-route** **Example:** `switch(config-lsp-attr)# record-route`	(Optional) Records the route used by the LSP.
Step 12	**no** *sub-command* **Example:** `switch(config-lsp-attr)# no record-route`	(Optional) Removes a specific attribute from the LSP attributes list. The *sub-command* argument names the LSP attribute to remove from the attributes list.
Step 13	**show mpls traffic-eng lsp attributes** [*name*] **Example:** `switch(config)# show mpls traffic-eng lsp attributes`	(Optional) Displays information about configured LSP attribute lists. Use the **show mpls traffic-eng lsp attributes** command to verify that the LSP attribute list was deleted from the switch.
Step 14	`show running mpls traffic-eng` **Example:** `switch(config)# show running mpls traffic-eng`	(Optional) Displays information about the running configuration of the MPLS TE feature.
Step 15	**show mpls traffic-eng tunnels tunnel-te** *number* **Example:** `switch(config)# show mpls traffic-eng tunnels tunnel-te 12`	(Optional) Displays the MPLS TE tunnels for the configured tunnel number.

	Command	Purpose
Step 16	**show running config interface tunnel-te** *number* **Example:** `switch(config)# show running config interface tunnel-te 12`	(Optional) Displays the running configuration of the interface MPLS TE tunnels for the configured tunnel number.
Step 17	**copy running-config startup-config** **Example:** `switch(config-lsp-attr)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Associating an LSP Attribute List with an MPLS TE Tunnel

You can associate an LSP attribute list with a path option for one or more MPLS TE tunnels.

Path option attributes for a TE tunnel are determined as follows:

- Attributes configured in an LSP attribute list on a path-option take precedence over the same attributes if they are configured directly in the tunnel interface configuration mode.
- If an attribute is not specified in the LSP attribute list, the device uses the attribute in the tunnel configuration. An LSP attribute list has no default values.
- If the attribute is not configured on the tunnel and not in the attribute list, the device uses the tunnel default attribute value. See the "Default Settings for LSP Attributes for MPLS TE" section on page 14-216.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **path-option** *number* {**dynamic** | **explicit** {**name** *path-name* | *path-number*} [**verbatim**]} [**attributes** *string*] [**bandwidth** *kbps*] [**lockdown**]
4. (Optional) **show interface tunnel-te** *number*
5. (Optional) **show mpls traffic-eng tunnels tunnel-te** *number*
6. (Optional) **show running config interface tunnel-te** *number*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**path-option** *number* {**dynamic** | **explicit** {**name** *path-name* | *path-number*} [**verbatim**]} [**attributes** *string*] [**bandwidth** *kbps*] [**lockdown**] **Example:** `switch(config-if-te)# path-option 1 dynamic attributes 1`	Adds an LSP attribute list to specify LSP-related parameters for path options for an MPLS TE tunnel. The arguments are as follows: • The *number* argument identifies the path option. • The **dynamic** keyword indicates that the path option is dynamically calculated (the switch figures out the best path). • The **explicit** keyword indicates that the path option is specified. You specify the IP addresses of the path. • The **name** *path-name* keyword argument combination identifies the name of the explicit path option. • The *path-number* argument identifies the number of the explicit path option. • The **verbatim** keyword bypasses the topology database verification. **Note** You can use the **verbatim** keyword only with the explicit path option. • The **attributes** *string* keyword argument combination names an attribute list to specify path options for the LSP. • The **bandwidth** keyword specifies LSP bandwidth. • The *kbps* argument is the number of kilobits per second set aside for the tunnel when signaled with this path option. The range is from 1 to 4294967295. • The **lockdown** keyword disables reoptimization of the LSP when the tunnel is signaled with this path option.
Step 4	**show interface tunnel-te** *number* **Example:** `switch(config-if-te)# show interface tunnel-te 1`	(Optional) Displays information about the TE tunnel.

	Command	Purpose
Step 5	**show mpls traffic-eng tunnels tunnel-te** *number* **Example:** `switch(config-if-te)# show mpls traffic-eng tunnels tunnel-te 12`	(Optional) Displays the MPLS TE tunnels for TE for the configured tunnel number.
Step 6	**show running config interface tunnel-te** *number* **Example:** `switch(config-if-te)# show running config interface tunnel-te 12`	(Optional) Displays the running configuration of the interface MPLS TE tunnels for the configured tunnel number.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if-te)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring a Path Option for Bandwidth Override

You can configure fallback bandwidth path options for a TE tunnel using the bandwidth parameter in the path option. You can configure path options that reduce the bandwidth constraint each time the headend of a tunnel fails to establish an LSP.

The bandwidth path option override is a temporary reduction in the bandwidth constraint. You can also use this feature to increase the bandwidth requirement. To force immediate reoptimization of all TE tunnels, use the global **reoptimize** command. You can also configure the **lockdown** command with the bandwidth override to prevent automatic reoptimization.

Note Once you configure the bandwidth as a path-option parameter, you can no longer configure an LSP attribute list as a path-option parameter.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **path-option** *number* {**dynamic** | **explicit** {**name** *path--name* | *path-number*} [**verbatim**]} [**attributes** *string*] [**bandwidth** *kbps*] [**lockdown**]
4. (Optional) **show interface tunnel-te** *number*
5. (Optional) **show mpls traffic-eng tunnels tunnel-te** *number*
6. (Optional) **show running config interface tunnel-te** *number*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**path-option** *number* {**dynamic** | **explicit** {**name** *path-name* | *path-number*} [**verbatim**]} [**attributes** *string*] [**bandwidth** *kbps*] [**lockdown**] **Example:** `switch(config-if-te)# path-option 1 dynamic attributes 1`	Adds an LSP attribute list to specify LSP-related parameters for path options for an MPLS TE tunnel. The arguments are as follows: • The *number* argument identifies the path option. • The **dynamic** keyword indicates that the path option is dynamically calculated (the switch figures out the best path). • The **explicit** keyword indicates that the path option is specified. You specify the IP addresses of the path. • The **name** *path-name* keyword argument combination identifies the name of the explicit path option. • The *path-number* argument identifies the number of the explicit path option. • The **verbatim** keyword bypasses the topology database verification. **Note** You can use the **verbatim** keyword only with the explicit path option. • The **attributes** *string* keyword argument combination names an attribute list to specify path options for the LSP. • The **bandwidth** keyword specifies the LSP bandwidth. • The *kbps* argument is the number of kilobits per second set aside for the path option. The range is from 1 to 4294967295. • The **lockdown** keyword disables reoptimization of the LSP.
Step 4	**show interface tunnel-te** *number* **Example:** `switch(config-if-te)# show interface tunnel-te 1`	(Optional) Displays information about the TE tunnel.

	Command	Purpose
Step 5	**show mpls traffic-eng tunnels tunnel-te** *number* **Example:** `switch(config-if-te)# show mpls traffic-eng tunnels tunnel-te 1`	(Optional) Displays the MPLS TE tunnels for the configured tunnel number.
Step 6	**show running config interface tunnel-te** *number* **Example:** `switch(config-if-te)# show running config interface tunnel-te 1`	(Optional) Displays the running configuration of the interface MPLS TE tunnels for the configured tunnel *number*.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if-te)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the Configuration for LSP Attributes for MPLS TE

To display the MPLS TE configuration for LSP attributes, perform one of the following tasks:

Command	Purpose
show mpls traffic-eng lsp attributes [*string*] [**details**]	Displays information about LSP attribute lists.
show mpls traffic-eng tunnels *tunnel-interface* [**brief**]	Displays information about MPLS TE tunnel attributes and path options.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Configuration Examples for LSP Attributes for MPLS TE

This section includes the following topics:

Example: LSP Attribute List on a TE Tunnel

The following example shows how to configure an LSP attribute list and associate it with an MPLS TE tunnel:

```
configuration terminal
feature mpls traffic-engineering
mpls traffic-eng configuration
lsp attributes 3
 affinity 0x3 mask 0xff
```

```
 bandwidth 1000
 priority 1 1

interface Tunnel-te 1
 ip unnumbered Ethernet2/1
 destination 10.112.0.12
 affinity 1
 bandwidth 5000
 path-option 1 dynamic attributes 3
```

Example: Path Option for Bandwidth Override

The following example shows how to configure a path option to override the bandwidth:

Note Once you configure the bandwidth as a path-option parameter, you can no longer configure an LSP attribute list as a path-option parameter.

```
configuration terminal
feature mpls traffic-engineering
interface Tunnel-te 1
 ip unnumbered Loopback0
 destination 10.10.10.12
 autoroute announce
 priority 1 1
 bandwidth 1000
 path-option 1 explicit name path1
 path-option 2 explicit name path2 bandwidth 500
 path-option 3 dynamic bandwidth 0
```

The device selects a path option for an LSP in order of preference, as follows:

- The device tries to signal an LSP using path options starting with path option 1.

  The device tries to signal an LSP with the 1000-kbps bandwidth configured on the tunnel interface because path option 1 has no bandwidth configured.

- If 1000 kbps is not available, the device tries to establish an LSP using path option 2.

  Path option 2 has 500 kbps configured, which reduces the bandwidth constraint from the original 1000-kbps configured on the tunnel interface.

- If 500 kbps is not available, the device tries to establish an LSP using path option 3.

  Path option 3 is configured as dynamic and has a bandwidth of 0. The device establishes the LSP if an MPLS TE path exists to the destination and all other tunnel constraints are met.

  If explicit path option 1 and explicit path option 2 both fail, dynamic path option 3 is attempted with 0 bandwidth, so it should succeed if any path exists. This option is a fallback to best effort.

Additional References for MPLS TE

The following sections provide references related to the LSP Attributes feature.

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download Cisco MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for LSP Attributes for MPLS TE

Table 14-2 lists the release history for this feature.

Table 14-2 Feature History for LSP Attributes for MPLS TE Tunnels

Feature Name	Releases	Feature Information
LSP attributes for MPLS TE tunnels	5.2(1)	This feature was introduced.

Configuring MPLS TE Verbatim Paths

This chapter describes how to configure a Multiprotocol Label Switching (MPLS) traffic engineering (TE) verbatim path on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS TE Verbatim Paths

Verbatim paths allow network nodes to support Resource Reservation Protocol (RSVP) extensions without supporting Interior Gateway Protocol (IGP) extensions for TE, thereby bypassing the topology database verification process.

MPLS TE label switched paths (LSPs) usually require that all the nodes in the network are TE aware, which means that they have IGP extensions to TE in place. However, some network administrators want to be able to build TE LSPs to traverse nodes that do not support IGP extensions to TE but that do support RSVP extensions to TE.

Verbatim LSPs are helpful when all or some of the intermediate nodes in a network do not support IGP extensions for TE.

When you enable a verbatim path, the IP explicit path is not checked against the TE topology database. Because the TE topology database is not verified, a message with IP explicit path information is routed using the shortest path first (SPF) algorithm for IP routing.

Licensing Requirements for MPLS TE Verbatim Paths

Product	License Requirement
Cisco NX-OS	Verbatim paths require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS TE Verbatim Paths

Verbatim paths have the following prerequisites:

- You must enable the MPLS TE feature. See the "Configuring MPLS TE" section on page 10-139.
- You must globally configure an MPLS TE tunnel.
- You must enable MPLS TE on all links that will be used for LSPs.

Guidelines and Limitations for MPLS TE Verbatim Paths

Verbatim paths have the following configuration guidelines and limitations:

- You can only use verbatim paths on an LSP that is configured with the explicit path option.
- Verbatim LSPs do not support reoptimization.
- You cannot configure MPLS TE over the logical generic routing encapsulation (GRE) tunnel interface.

Configuring MPLS TE Verbatim Paths

You can configure verbatim paths on MPLS TE tunnels by using the verbatim path option.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*

3. **path-option** *number* {**dynamic** | **explicit** {**name** *path-name* | *path-number*} [**verbatim**]} [**attributes** *string*] [**bandwidth** *kbps*] [**lockdown**]}
4. (Optional) **show interface tunnel-te** *number*
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** switch(config)# interface tunnel-te 1	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**path-option** *number* {**dynamic** | **explicit** {**name** *path-name* | *path-number*} [**verbatim**]} [**attributes** *string*] [**bandwidth** *kbps*] [**lockdown**] **Example:** switch(config-if-te)# path-option 1 dynamic attributes 1	Adds an LSP attribute list to specify LSP-related parameters for path options for an MPLS TE tunnel. The arguments are as follows: • The *number* argument identifies the path option. • The **dynamic** keyword indicates that the path option is dynamically calculated (the switch figures out the best path). • The **explicit** keyword indicates that the path option is specified. You specify the IP addresses of the path. • The **name** *path-name* keyword argument combination identifies the name of the explicit path option. • The *path-number* argument identifies the number of the explicit path option. • The **verbatim** keyword bypasses the topology database verification. **Note** You can use the **verbatim** keyword only with the explicit path option. • The **attributes** *string* keyword argument combination names an attribute list to specify path options for the LSP. • The **bandwidth** keyword specifies LSP bandwidth. • The *kbps* argument is the number of kilobits per second set aside for the path option. The range is from 1 to 4294967295. • The **lockdown** keyword disables reoptimization of the LSP.

	Command	Purpose
Step 4	**show interface tunnel-te** *number* **Example:** `switch(config-if-te)# show interface tunnel-te 1`	(Optional) Displays information about the TE tunnel.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if-te)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS TE Verbatim Path Configuration

To display the MPLS TE verbatim path configuration, perform the following task:

Command	Purpose
show mpls traffic-eng tunnels *tunnel-interface* [**brief**]	Displays information about MPLS TE tunnel attributes and path options.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Configuration Example for MPLS TE Verbatim Paths

This section provides the following configuration example:

- Example: Verbatim Path, page 15-233

Example: Verbatim Path

The following example shows how to configure a tunnel with an explicit path option using verbatim paths:

```
configuration terminal
feature mpls traffic-engineering
interface tunnel-te 1
 ip unnumbered loopback 1
 destination 10.10.100.100
 bandwidth 1000
 path-option 1 explicit name path1 verbatim
 no shutdown
```

Additional References for MPLS TE Verbatim Paths

The following sections provide references related to the verbatim path feature.

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download Cisco MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS TE Verbatim Paths

Table 15-1 lists the release history for this feature.

Table 15-1 Feature History for Verbatim Path

Feature Name	Releases	Feature Information
Verbatim path	5.2(1)	This feature was introduced.

Configuring MPLS TE Forwarding Adjacency

This chapter describes how to configure Multiprotocol Label Switching (MPLS) traffic engineering (TE) forwarding adjacency on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

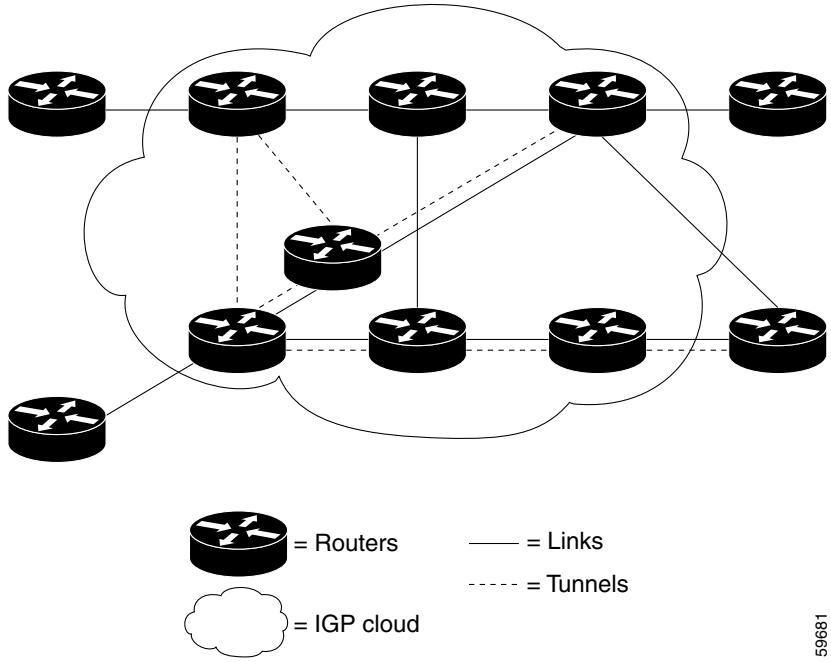
Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS TE Forwarding Adjacency

MPLS TE forwarding adjacency allows you to handle a TE label-switched path (LSP) tunnel as a link in an Interior Gateway Protocol (IGP) network that is based on the Shortest Path First (SPF) algorithm. Both Intermediate System-to-Intermediate System (IS-IS) and Open Shortest Path First (OSPF) are supported as the IGP.

A forwarding adjacency can be created between switches regardless of their location in the network. The switches can be located multiple hops from each other, as shown in Figure 16-1.

Figure 16-1 Forwarding Adjacency Topology



As a result, a TE tunnel is advertised as a link in an IGP network with the tunnel's cost associated with it.

Switches outside of the TE domain see the TE tunnel and use it to compute the shortest path for routing traffic throughout the network.

TE tunnel interfaces are advertised in the IGP network just like any other links. Switches can then use these advertisements in their IGPs to compute the SPF even if they are not the headend of any TE tunnels.

Licensing Requirements for MPLS TE Forwarding Adjacency

Product	License Requirement
Cisco NX-OS	Forwarding adjacency requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS TE Forwarding Adjacency

Forwarding adjacency has the following prerequisites:

- You must enable the MPLS TE feature. See the "Configuring MPLS TE" section on page 10-139.

Guidelines and Limitations for MPLS TE Forwarding Adjacency

Forwarding adjacency has the following configuration guidelines and limitations:

- Forwarding adjacency increases the size of the IGP database by advertising a TE tunnel as a link.
- When you enable forwarding adjacency on a TE tunnel, the link is advertised in the IGP network as a type, length, value (TLV) 22 object without any TE sub-TLV.
- You must configure MPLS TE forwarding adjacency tunnels bidirectionally.
- You cannot configure MPLS TE over the logical generic routing encapsulation (GRE) tunnel interface.

Note If both forwarding adjacency and autoroute announce are configured on a link, forwarding adjacency takes precedence. The autoroute configuration takes effect automatically if the forwarding adjacency configuration is removed, but not if the forwarding adjacency path fails.

Default Settings for MPLS TE Forwarding Adjacency

Table 16-1 lists the default settings for forwarding adjacency.

Table 16-1 Default Settings for Forwarding Adjacency

Parameters	Default
forwarding-adjacency holdtime	Defaults to 0

Configuring MPLS TE Forwarding Adjacency

You can configure a tunnel interface for an MPLS TE forwarding adjacency.

Note You must configure a forwarding adjacency on two LSP tunnels bidirectionally from A to B and B to A. Otherwise, the forwarding adjacency is advertised but not used in the IGP network.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **forwarding-adjacency** [**holdtime** *value*]
4. **isis metric** *metric-value* {**level-1** | **level-2**}
5. (Optional) **show interface tunnel-te** *number*

6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**forwarding-adjacency** [**holdtime** *value*] **Example:** `switch(config-if-te)#` `forwarding-adjacency`	Advertises a TE tunnel as a link in an IGP network. The holdtime specifies the time (in milliseconds) that TE waits after the tunnel goes down before informing the IGP. This process enables the tunnel to try to find an alternate path within that time frame and if successful, the IGP is never notified that the tunnel went down (which avoids unnecessary SPF calculations).
Step 4	**isis metric** *metric-value* {**level-1** | **level-2**} **Example:** `switch(config-if-te)# isis metric 2` `level-1`	Configures the IS-IS metric for a tunnel interface to be used as a forwarding adjacency. You should specify the **isis metric** command with level-1 or level-2 to be consistent with the IGP level at which you are performing TE. Otherwise, the metric has the default value of 10. **Note** Use this command only if the IGP is IS-IS. If the IGP is OSPF, use the equivalent OSPF command.
Step 5	**show interface tunnel-te** *number* **Example:** `switch(config-if-te)# show interface` `tunnel-te 1`	(Optional) Displays information about the TE tunnel.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if-te)# copy` `running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS TE Forwarding Adjacency Configuration

To display the MPLS TE forwarding adjacency configuration, perform one of the following tasks:

Command	Purpose
show mpls traffic-eng forwarding-adjacency [*ip-address*]	Displays information about MPLS TE forwarding adjacency.
show isis [*process-tag*] **database** [**level-1**] [**level-2**] [**l1**] [**l2**] [**detail**] [**lspid**]	Displays information about IS-IS.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Configuration Example for MPLS TE Forwarding Adjacency

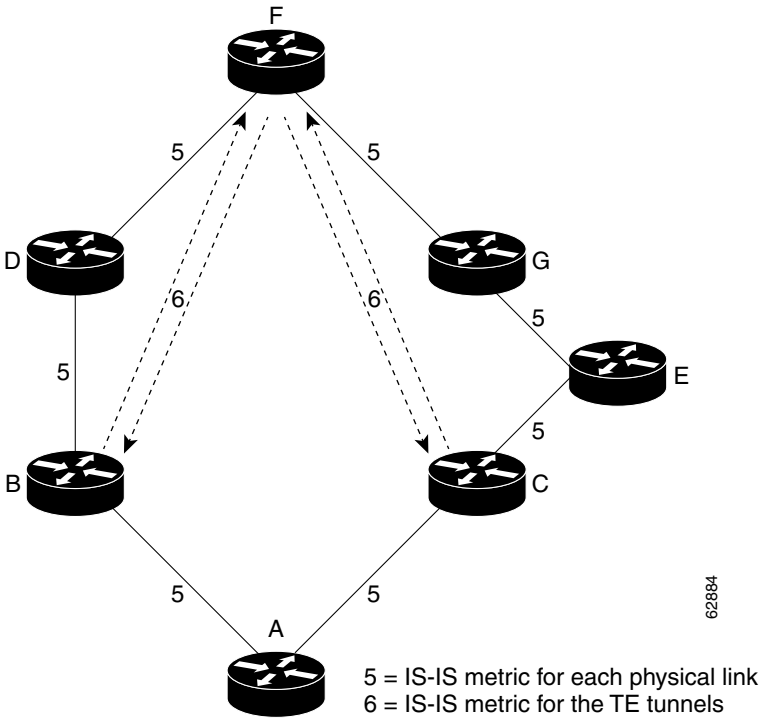
The following example shows how to configure a TE tunnel interface with forwarding adjacency and an IS-IS metric:

```
configuration terminal
feature mpls traffic-engineering
interface tunnel-te 7
 forwarding-adjacency
 isis metric 2 level-1
```

Note If both the **forwarding adjacency** command and the **autoroute announce** command appear in your configuration, the **forwarding adjacency** command takes precedence.

In Figure 16-2, if you have no forwarding adjacencies configured for the TE tunnels between B and F, and C and F, all the traffic that A must forward to F goes through B because B is the shortest path from A to F. (The cost from A to F is 15 through B and 20 through C.)

Figure 16-2 Using Forwarding Adjacencies



If you have forwarding adjacencies configured on the TE tunnels between B and F and C and F and also on the TE tunnels between F and B, and F and C, then when A computes the SPF algorithm, A sees two equal cost paths of 11 to F. As a result, traffic across the A–B and A–C links is shared.

Additional References for MPLS TE Forwarding Adjacency

For additional information related to the Forwarding Adjacency feature, see the following sections:

- Related Documents, page 16-241
- MIBs, page 16-241

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download Cisco MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for MPLS TE Forwarding Adjacency

Table 16-2 lists the release history for this feature.

Table 16-2 Feature History for Forwarding Adjacency

Feature Name	Releases	Feature Information
MPLS TE forwarding adjacency	5.2(1)	This feature was introduced.

Configuring MPLS TE Path Protection

This chapter describes how to configure Multiprotocol Label Switching (MPLS) path protection for traffic engineering (TE) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS TE Path Protection

Path protection provides an end-to-end failure recovery function (full path protection) for MPLS TE tunnels. Cisco NX-OS supports regular path protection and enhanced path protection, which is the ability to configure up to eight secondary path options for a given primary path option.

This section includes the following topics:

Path Protection

A secondary label switched path (LSP) is configured and established to provide failure protection for the LSP that is carrying a tunnel's TE traffic. When there is a failure on the protected LSP, the headend router immediately enables the secondary LSP to temporarily carry the tunnel's traffic. If there is a failure on the secondary LSP, the tunnel no longer has path protection until the failure along the secondary path is cleared.

Path protection can be used with

- Single area
  - Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF)
- Interarea
  - Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF)
- Inter-AS
  - Border Gateway Protocol (BGP), both external BGP (eBGP,) and static

The failure detection functions that trigger a switchover to a secondary tunnel include the following:

- Path error or resv tear from RSVP signaling
- Notification from the Bidirectional Forwarding Detection (BFD) protocol that a neighbor is lost
- Notification from the Interior Gateway Protocol (IGP) that the adjacency is down
- Local teardown of the protected tunnel's LSP due to preemption in order to signal higher priority LSPs, online insertion and removal (OIR), and so forth

An alternate recovery function is Fast Reroute (FRR), which protects MPLS TE LSPs only from link and node failures by locally repairing the LSPs at the point of failure.

Although not as fast as link or node protection, presignaling a secondary LSP is faster than configuring a secondary primary path option or allowing the tunnel's headend router to dynamically recalculate a path. The actual recovery time is topology-dependent and is affected by delay factors such as propagation delay or switch fabric latency.

Enhanced Path Protection

Enhanced path protection provides support of multiple backup path options per primary path option. You can configure up to eight backup path options for a given primary path option. Only one of the configured backup path options is actively signaled at any time.

After you enter the **mpls traffic-eng path-option list** command, you can enter the backup path priority in the *number* argument of the **path-option** command. A lower identifier represents a higher priority. Priorities are configurable for each backup path option. Multiple backup path options and a single backup path option cannot coexist to protect a primary path option.

ISSU

Cisco In Service Software Upgrade (ISSU) allows you to perform a Cisco NX-OS software upgrade or downgrade while the system continues to forward packets. ISSU takes advantage of the Cisco NX-OS high availability infrastructure (Cisco nonstop forwarding [NSF] with stateful switchover [SSO] and hardware redundancy) and eliminates the downtime that is associated with software upgrades or version changes by allowing changes while the system remains in service. Cisco ISSU lowers the impact that planned maintenance activities have on network service availability; there is less downtime and better access to critical systems.

When path protection is enabled and an ISSU upgrade is performed, path protection performance is similar to that of other TE features.

NSF/SSO

Cisco NSF with SSO provides continuous packet forwarding, even during a network processor hardware or software failure.

Path protection can recover after SSO. A tunnel configured for path protection may have two LSPs signaled simultaneously: the primary LSP that is carrying the traffic and the secondary LSP that carries traffic if there is a failure along the primary path. Only information associated with one of those LSPs, the one that is currently carrying traffic, is synchronized to the standby system. On recovery, the standby system can determine from the checkpointed information whether the LSP was the primary or secondary.

If the primary LSP was active during the switchover, only the primary LSP is recovered. The secondary LSP that was signaled and provided path protection is resignaled after the TE recovery period is complete. This process does not impact the traffic on the tunnel because the secondary LSP was not carrying traffic.

Licensing Requirements for MPLS TE Path Protection

Product	License Requirement
Cisco NX-OS	MPLS TE path protection requires an MPLS license. For a complete explanation of the NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS TE Path Protection

Path protection has the following prerequisites:

- The MPLS TE feature must be enabled. MPLS TE can be enabled or disabled by the [**no**] **feature mpls traffic-eng** command and is disabled by default. See the "Configuring MPLS TE" section on page 10-139.
- Configure a TE tunnel with a primary path option by using the **path-option** command.

Guidelines and Limitations for MPLS TE Path Protection

Path protection has the following configuration guidelines and limitations:

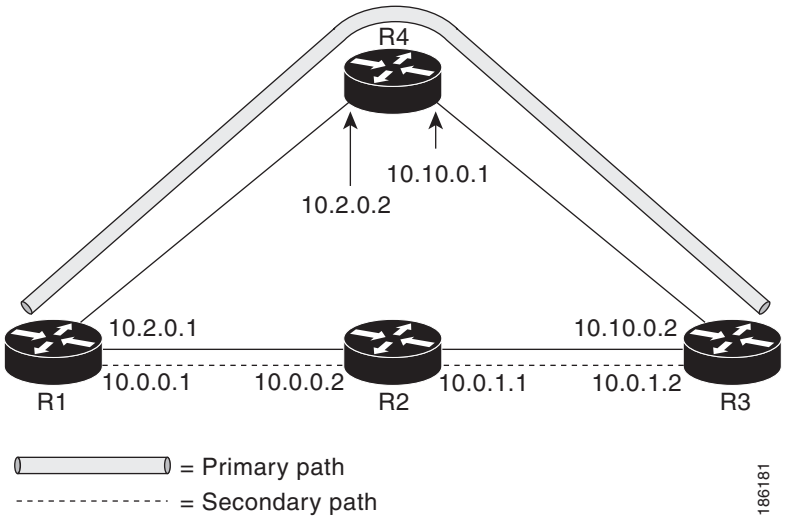
- The secondary path will not be signaled with the FRR flag.
- Dynamic diverse paths are not supported. You must configure an explicit path for the secondary LSP that avoids using any shared links with the primary LSP.
- Do not use link and node protection with path protection on the headend router.

Configuring MPLS TE Path Protection

This section includes the following topics:

These tasks are described in the following sections and are shown in Figure 17-1.

Figure 17-1 Network Topology—Path Protection



Configuring Explicit Paths for Secondary Paths

You can specify a secondary path that does not include common links or nodes associated with the primary path in case those links or nodes go down.

Prerequisites

You must enable the MPLS TE feature (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng configuration**
3. [**no**] **explicit-path** {**name** *path-name* | **identifier** *number*}
4. **index** *index command ip-address*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls traffic-eng configuration** **Example:** `switch(config)# mpls traffic-eng configuration` `switch(config-te)#`	Enters MPLS TE configuration mode.
Step 3	[**no**] **explicit-path** {**name** *path-name* | **identifier** *number*} **Example:** `switch(config-te)# explicit-path name path1223` `switch(config-te-expl-path)`	Creates or modifies the explicit path and enters explicit path configuration mode.
Step 4	**index** *index command ip-address* **Example:** `switch(config-te-expl-path)# index 10 next-address 10.0.0.2`	Inserts or modifies a path entry at a specific index. The *command* argument can be the **exclude-address** keyword or the **next-address** keyword. The *ip-address* argument represents the node ID. **Note** Enter this command once for each router or switch along the secondary path.

Assigning a Secondary Path Option to Protect a Primary Path Option

You can assign a secondary path option in case there is a link or node failure along a path and all interfaces in your network are not protected.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*

3. **path-option protect** *number* **explicit** {**identifier** *path-number* | **name** *path-name*} [**attributes** *lsp-attributes* | **bandwidth** *kbps* | **lockdown**] [**verbatim**]

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1` `switch(config-if-te)#`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**path-option protect** *number* **explicit** {**identifier** *path-number* | **name** *path-name*} [**attributes** *lsp-attributes* | **bandwidth** *kbps* | **lockdown**] [**verbatim**] **Example:** `switch(config-if-te)# path-option protect 10 explicit name path344`	Configures a secondary path option for an MPLS TE tunnel.

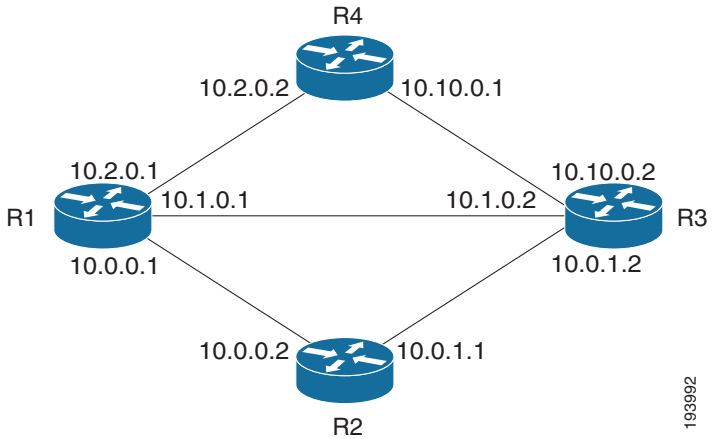
Enhanced Path Protection Configuration Tasks

This section includes the following topics:

- Creating a Path Option List, page 17-250
- Assigning a Path Option List to Protect a Primary Path Option, page 17-251

These tasks are described in the following sections and are shown in Figure 17-2.

Figure 17-2 Network Topology - Enhanced Path Protection



Creating a Path Option List

In enhanced path protection you can create and assign a path option list.

Note To use a secondary path instead, follow the steps in the "Configuring Explicit Paths for Secondary Paths" section on page 17-247.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **mpls traffic-eng path-option list** [**name** *pathlist-name* | **identifier** *pathlist-number*]
4. **path-option** *number* **explicit [name** *pathoption-name* **| identifier** *pathoption-number*]
5. **list**
6. **no** [*pathoption-name* | *pathoption-number*]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1` `switch(config-if-te)#`	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**mpls traffic-eng path-option list** [**name** *pathlist-name* | **identifier** *pathlist-number*] **Example:** `switch(config)# mpls traffic-eng path-option list name pathlist-01`	Configures a path option list, and enters path-option list configuration mode. You can enter the following commands: **path-option**, **list**, **no**, and **exit**.
Step 4	**path-option** *number* **explicit [name** *pathoption-name* **| identifier** *pathoption-number***]** **Example:** `switch(cfg-pathoption-list)# path-option 10 explicit identifier 200`	(Optional) Specifies the name or identification number of the path option to add, edit, or delete. The *pathoption-number* value can be from 1 through 65535.
Step 5	**list** **Example:** `switch(cfg-pathoption-list)# list`	(Optional) Lists all of the path options.

	Command or Action	Purpose
Step 6	**no** [*pathoption-name* | *pathoption-number*] **Example:** switch(cfg-pathoption-list)# no 10	(Optional) Deletes a specified path option.
Step 7	**end** **Example:** switch(cfg-pathoption-list)# exit switch#	Exits to EXEC mode.

Assigning a Path Option List to Protect a Primary Path Option

You can assign a path option list in case there is a link or node failure along a path and all interfaces in your network are not protected. See Figure 17-2.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **path-option protect** *number* [**attributes** *lsp-attributes* | **bandwidth** {*kbps* | **subpool** *kbps*} | **explicit** {**identifier** *path-number* | **name** *path-name*} | **list** {**name** *pathlist-name* | **identifier** *pathlist-identifier*}]
4. **end**

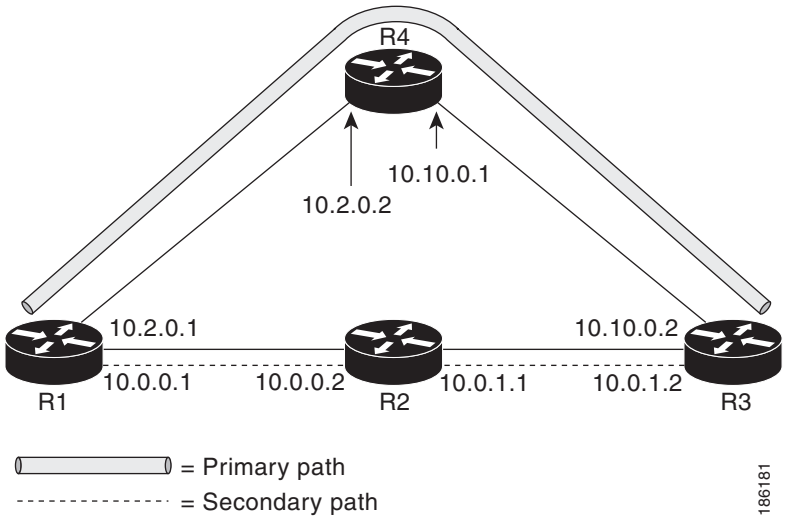
DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** switch(config)# interface tunnel-te 500 switch(config-if-te)#	Enters TE interface configuration mode. The *number* argument identifies the tunnel number to be configured.
Step 3	**path-option protect** *number* [**attributes** *lsp-attributes* | **bandwidth** {*kbps* | **subpool** *kbps*} | **explicit** {**identifier** *path-number* | **name** *path-name*} | **list** {**name** *pathlist-name* | **identifier** *pathlist-identifier*}] **Example:** switch(config-if-te)# path-option protect 10 list name pathlist-01	Configures a path option list to protect primary path option 10.
Step 4	**end** **Example:** switch(config-if-te)# end switch#	Exits to EXEC mode.

Verifying the MPLS TE Path Protection Configuration

You can display the path protection configuration. Steps 1 and 2 refer to Figure 17-3.

Figure 17-3 Network Topology Verification



To display path protection for verification, perform one of the following tasks:

Command	Purpose
show running interface tunnel-te *number*	Displays the configuration of the primary path and the protection path options.
show mpls traffic-eng tunnels tunnel-te *number*	Displays tunnel path information.
show mpls traffic-eng tunnels tunnel-te *number* **protection**	Displays the status of both LSPs (primary path and protected path), when the **protection** keyword is specified. **Note** Deleting a primary path option has the same effect as shutting down a link. Traffic moves to the protected path.

Examples

The following example shows how to display the configuration of the primary path and protection path options.

Note To show the status of both LSPs (primary path and protected path), use the **show mpls traffic-eng tunnels** command with the **protection** keyword.

```
switch# show running interface tunnel-te500

Building configuration...

Current configuration : 497 bytes
```

```
!
interface Tunnel-te500
 ip unnumbered Loopback0
 destination 10.0.0.9
 autoroute announce
 priority 7 7
 bandwidth 100
 path-option 10 explicit name path344
 path-option 20 explicit name path345
 path-option protect 10 explicit name path3441
 path-option protect 20 explicit name path348
end
```

The following example shows how to display tunnel path information.

The command output shows no common links or nodes.

Note The Common Link(s) field shows the number of links shared by both primary and secondary paths, from the headend router to the tailend router.

The Common Node(s) field shows the number of nodes shared by both primary and secondary paths, excluding the headend and tailend routers.

```
switch# show mpls traffic-eng tunnels tunnel-te500

Name: R1_t500 (Tunnel-te500) Destination: 10.0.0.9
 Status:
  Admin: up Oper: up Path: valid Signalling: connected
  path option 10, type explicit path344 (Basis for Setup, path weight 20)
  path option 20, type explicit path345
  Path Protection: 0 Common Link(s), 0 Common Node(s)
  path protect option 10, type explicit path3441 (Basis for Protect, path weight 20)
  path protect option 20, type explicit path348

 Config Parameters:
  Bandwidth: 100 kb/s (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute: enabled LockDown: disabled Loadshare: 100 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
  State: explicit path option 10 is active
  BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet1/0, 16
 RSVP Signalling Info:
    Src 10.1.1.1, Dst 10.0.0.9, Tun_Id 500, Tun_Instance 19
  RSVP Path Info:
    My Address: 10.2.0.1
    Explicit Route: 10.2.0.2  10.10.0.1  10.10.0.2  10.0.0.9
    Record Route: NONE
    Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  RSVP Resv Info:
    Record Route: NONE
    Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
 Shortest Unconstrained Path Info:
  Path Weight: 20 (TE)
  Explicit Route: 10.2.0.1  10.2.0.2  10.10.0.1  10.10.0.2  10.0.0.9
 History:
  Tunnel:
```

```
   Time since created: 11 minutes, 17 seconds
   Time since path change: 8 minutes, 5 seconds
   Number of LSP IDs (Tun_Instances) used: 19
  Current LSP:
   Uptime: 8 minutes, 5 seconds
```

The following example shows how to display the status of both LSPs (primary path and protected path) when the **protection** keyword is specified.

Note Deleting a primary path option has the same effect as shutting down a link. Traffic moves to the protected path.

The command output shows that both primary LSP and secondary LSP are up and protection is enabled:

```
switch# show mpls traffic-eng tunnels tunnel-te500 protection

R1_t500
 LSP Head, Tunnel-te500, Admin: up, Oper: up
 Src 10.1.1.1, Dest 10.0.0.9, Instance 19
 Fast Reroute Protection: None
 Path Protection: 0 Common Link(s), 0 Common Node(s)
  Primary lsp path:10.2.0.1  10.2.0.2
                   10.10.0.1 10.10.0.2
                   10.0.0.9
  Protect lsp path:10.0.0.1  10.0.0.2
                   10.0.1.1  10.0.1.2
                   10.0.0.9
  Path Protect Parameters:
   Bandwidth: 100 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
   Metric Type: TE (default)
  InLabel : -
  OutLabel : Ethernet0/0, 16
  RSVP Signalling Info:
    Src 10.1.1.1, Dst 10.0.0.9, Tun_Id 500, Tun_Instance 27
   RSVP Path Info:
    My Address: 10.0.0.1
    Explicit Route: 10.0.0.2  10.0.1.1  10.0.1.2  10.0.0.9
    Record Route: NONE
    Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
   RSVP Resv Info:
    Record Route: NONE
    Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
```

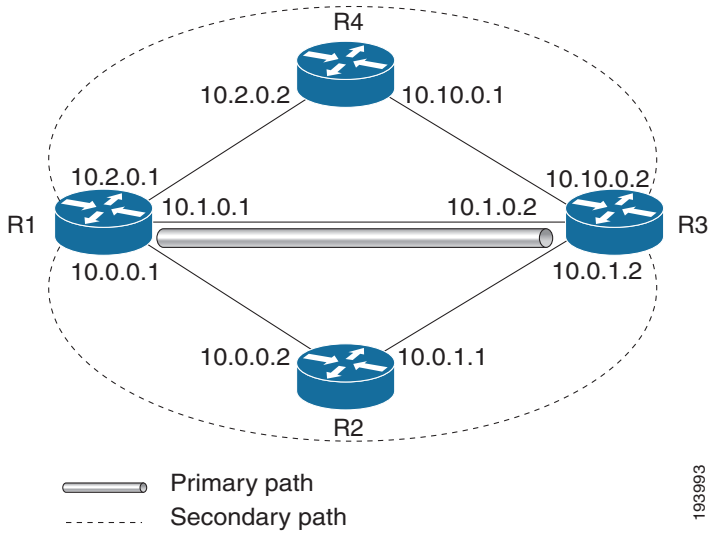
The following example shows that the primary LSP is down and that the secondary LSP is up and is carrying traffic:

```
switch# show mpls traffic-eng tunnels tunnel-te500 protection

R1_t500
 LSP Head, Tunnel-te500, Admin: up, Oper: up
 Src 10.1.1.1, Dest 10.0.0.9, Instance 27
 Fast Reroute Protection: None
 Path Protection: Backup lsp in use.
```

Verifying the Enhanced Path Protection Configuration

To verify the enhanced path protection configuration, refer to Figure 17-4 and perform the following steps.

Figure 17-4 Network Topology Verification for Enhanced Path Protection



SUMMARY STEPS

1. **show running interface tunnel-te** *number*
2. **show mpls traffic-eng tunnels tunnel-te** *number*
3. **show mpls traffic-eng tunnels tunnel-te** *number* [**brief**] [**protection**]
4. **show ip rsvp high-availability database** {**hello** | **link-management** {**interfaces** [**fixed** | **variable**] | **system**} | **lsp** [**filter destination** *ip-address* | **filter lsp-id** *lsp-id* | **filter source** *ip-address* | **filter tunnel-id** *tunnel-id*] | **lsp-head** [**filter** *number*] | **summary**}

DETAILED STEPS

Step 1 **show running interface tunnel-te** *number*

This command shows the configuration of the path option and backup path option.

Note To show the status of both LSPs (that is, both the primary path and the protected path), use the **show mpls traffic-eng tunnels** command with the **protection** keyword.

```
switch# show running interface tunnel-te2


Building configuration..

Current configuration : 296 bytes
!
interface Tunnel-te2
 ip unnumbered Loopback0
 destination 10.10.0.2
 autoroute announce
 path-option 10 explicit name primary1
```

```
 path-option protect 10 list name pathlist-01
end
```

Step 2 **show mpls traffic-eng tunnels tunnel-te** *number*

This command shows tunnel path information.

The Common Link(s) field shows the number of links shared by both the primary and secondary paths, from the headend router to the tailend router.

The Common Node(s) field shows the number of nodes shared by both the primary and secondary paths, excluding the headend and tailend routers.

As shown in the following output, there are no common links or nodes:

```
switch# show mpls traffic-eng tunnels tunnel-te2

Name: iou-100_t2 (Tunnel-te2) Destination: 10.10.0.2
 Status:
  Admin: up Oper: up Path: valid Signalling: connected
  path option 10, type explicit primary1 (Basis for Setup, path weight 10)
  Path Protection: 0 Common Link(s), 0 Common Node(s)
  path protect option 10, type list name secondary-list
   Inuse path-option 10, type explicit secondary1 (Basis for Protect, path weight 20)

Config Parameters:
 Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
 Metric Type: TE (default)
 AutoRoute announce: enabled LockDown: disabled Loadshare: 0 bw-based
 auto-bw: disabled
Active Path Option Parameters:
 State: explicit path option 10 is active
 BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

InLabel : -
OutLabel : Ethernet7/0, implicit-null
RSVP Signalling Info:
 Src 100.100.100.100, Dst 10.10.0.2, Tun_Id 2, Tun_Instance 188
RSVP Path Info:
 My Address: 10.1.0.1
 Explicit Route: 10.1.0.2 10.10.0.2
 Record Route: NONE
 Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
RSVP Resv Info:
 Record Route: NONE
 Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
Shortest Unconstrained Path Info:
 Path Weight: 10 (TE)
 Explicit Route: 10.1.0.1 10.1.0.2 10.10.0.2
History:
 Tunnel:
  Time since created: 1 hours, 34 minutes
  Time since path change: 1 minutes, 50 seconds
  Number of LSP IDs (Tun_Instances) used: 188
 Current LSP:
  Uptime: 1 minutes, 50 seconds
 Prior LSP:
  ID: path option 10 [44]
  Removal Trigger: label reservation removed
```

Step 3 **show mpls traffic-eng tunnels tunnel-te** *number* [**brief**] [**protection**]

Use this command, with the **protection** keyword specified, to show the status of both LSPs (that is, both the primary path and the protected path).

The following command output shows that the primary LSP is up, and the secondary LSP is up and providing protection:

```
switch# show mpls traffic-eng tunnels tunnel-te2 protection

iou-100_t2
 LSP Head, Tunnel-te2, Admin: up, Oper: up
 Src 100.100.100.100, Dest 10.10.0.2, Instance 188
 Fast Reroute Protection: None
 Path Protection: 0 Common Link(s), 0 Common Node(s)
  Primary lsp path:10.1.0.1 10.1.0.2
                   10.10.0.2
  Protect lsp path:10.0.0.1 10.0.0.2
                   10.0.1.1 10.0.1.2
                   10.10.0.2
  Path Protect Parameters:
   Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
   Metric Type: TE (default)
  InLabel : -
  OutLabel : Ethernet5/0, 16
  RSVP Signalling Info:
    Src 100.100.100.100, Dst 10.10.0.2, Tun_Id 2, Tun_Instance 189
   RSVP Path Info:
    My Address: 10.0.0.1
    Explicit Route: 10.0.0.2 10.0.1.1 10.0.1.2 10.10.0.2
    Record Route: NONE
    Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
   RSVP Resv Info:
    Record Route: NONE
    Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
```

The following command output shows that the primary LSP is down, and the secondary LSP is up and is actively carrying traffic:

```
switch# show mpls traffic-eng tunnels tunnel-te500 protection

R1_t500
 LSP Head, Tunnel-te500, Admin: up, Oper: up
 Src 10.1.1.1, Dest 10.0.0.9, Instance 27
 Fast Reroute Protection: None
 Path Protection: Backup lsp in use.
```

Step 4 **show ip rsvp high-availability database** {**hello** | **link-management** {**interfaces** [**fixed** | **variable**]} | **system**} | **lsp** [**filter destination** *ip-address* | **filter lsp-id** *lsp-id* | **filter source** *ip-address* | **filter tunnel-id** *tunnel-id*] | **lsp-head** [**filter** *number*] | **summary**}

The **show ip rsvp high-availability database** command displays the contents of the RSVP HA read and write databases used in TE. If you specify the **lsp-head** keyword, the command output includes path protection information.

```
switch# show ip rsvp high-availability database lsp-head

LSP_HEAD WRITE DB
 Tun ID: 2
 Header:
  State: Checkpointed Action: Add
  Seq #: 2 Flags: 0x0
 Data:
  lsp_id: 6, bandwidth: 0, thead_flags: 0x1, popt: 10
  feature flags: none
  output_if_num: 31, output_nhop: 10.1.0.2
 RRR path setup info
  Destination: 10.10.0.2, Id: .10.10.0.2 Router Node (ospf) flag:0x0
```

```
IGP: ospf, IGP area: 0, Number of hops: 3, metric: 10
Hop 0: 10.1.0.1, Id: 10.100.100.100 Router Node (ospf), flag:0x0
Hop 1: 10.1.0.2, Id: 10.10.0.2 Router Node (ospf), flag:0x0
Hop 2: 10.103.103.103, Id: 10.10.0.2 Router Node (ospf), flag:0x0
```

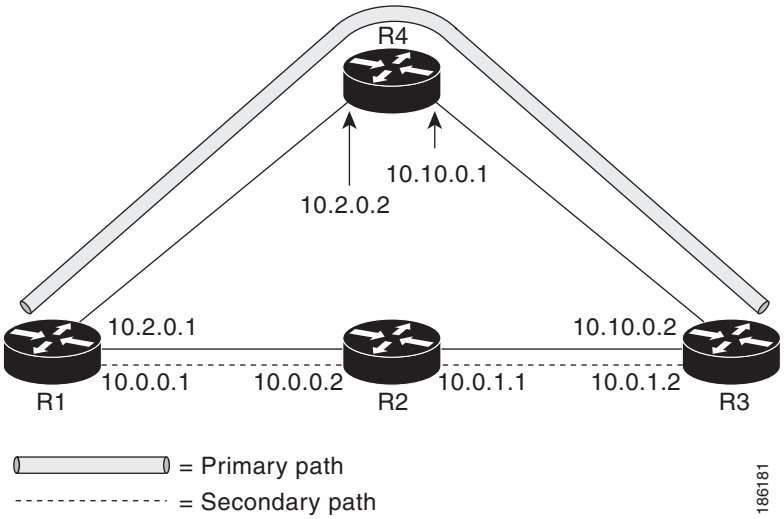
Configuration Examples for MPLS TE Path Protection

This section includes the following topics:

Example: Configuring Explicit Paths for Secondary Paths

Figure 17-5 shows a primary path and a secondary path. If there is a failure, the secondary path is used.

Figure 17-5 Primary Path and Secondary Path



The following example shows that the explicit path is named path3441. There is an **index** command for each router. If there is failure, the secondary path is used.

```
switch(config)# mpls traffic-eng configuration
switch(config-te)# explicit-path name path3441
switch(config-te-expl-path)# index 1 next-address 10.0.0.1
Explicit Path name path3441:
    1: next-address 10.0.0.1

switch(config-te-expl-path)# index 2 next-address 10.0.0.2
Explicit Path name path3441:
    1: next-address 10.0.0.1
    2: next-address 10.0.0.2
```

```
switch(config-te-expl-path)# index 3 next-address 10.0.1.1
Explicit Path name path3441:
    1: next-address 10.0.0.1
    2: next-address 10.0.0.2
    3: next-address 10.0.1.1

switch(config-te-expl-path)# index 4 next-address 10.0.1.2
Explicit Path name path3441:
    1: next-address 10.0.0.1
    2: next-address 10.0.0.2
    3: next-address 10.0.1.1
    4: next-address 10.0.1.2
```

Example: Assigning a Secondary Path Option to Protect a Primary Path Option

The following example shows that a TE tunnel is configured:

```
switch# configure terminal
switch(config)# interface tunnel-te500
switch(config-if-te)# path-option protect 10 explicit name path344
```

The following example shows that path protection has been configured. Tunnel 500 has path option 10 using path344 and protected by path 3441, and path option 20 using path345 and protected by path348.

```
switch# show running interface tunnel-te500

Building configuration...

Current configuration : 497 bytes
!
interface Tunnel-te500
 ip unnumbered Loopback0

 destination 10.0.0.9
 autoroute announce
 priority 7 7
 bandwidth 100
 path-option 10 explicit name path344
 path-option 20 explicit name path345
 path-option protect 10 explicit name path3441
 path-option protect 20 explicit name path348
end
```

Example: Configuring Tunnels Before and After Path Protection

The following example shows information about the primary (protected) path. The following sample output shows that path protection has been configured:

```
switch# show mpls traffic-eng tunnels tunnel-te500

Name: R1_t500   (Tunnel-te500)   Destination: 10.0.0.9
 Status:
  Admin: up  Oper: up  Path: valid  Signalling: connected
  path option 10, type explicit path344 (Basis for Setup, path weight 20)
  path option 20, type explicit path345
  Path Protection: 0 Common Link(s), 0 Common Node(s)
  path protect option 10, type explicit path3441 (Basis for Protect, path weight 20)
  path protect option 20, type explicit path348

 Config Parameters:
```

```
  Bandwidth: 100 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute: enabled LockDown: disabled Loadshare: 100 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
    State: explicit path option 10 is active
BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet1/0, 16
 RSVP Signalling Info:
   Src 10.1.1.1, Dst 10.0.0.9, Tun_Id 500, Tun_Instance 43
  RSVP Path Info:
   My Address: 10.2.0.1
   Explicit Route: 10.2.0.2  10.10.0.1  10.10.0.2  10.0.0.9
   Record Route: NONE
   Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  RSVP Resv Info:
   Record Route: NONE
   Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  Shortest Unconstrained Path Info:
   Path Weight: 20 (TE)
   Explicit Route: 10.0.0.1  10.0.0.2  10.0.1.1  10.0.1.2
                   10.0.0.9
 History:
  Tunnel:
   Time since created: 18 minutes, 22 seconds
   Time since path change: 19 seconds
   Number of LSP IDs (Tun_Instances) used: 43
  Current LSP:
   Uptime: 22 seconds
   Selection: reoptimization
  Prior LSP:
   ID: path option 10 [27]
   Removal Trigger: reoptimization completed
```

The following example shows information about the secondary path. Tunnel500 is protected. The protection path is used, and the primary path is down. The command output shows the IP explicit paths of the primary LSP and the secondary LSP.

```
switch# show mpls traffic-eng tunnels tunnel-te500 protection

R1_t500
 LSP Head, Tunnel500, Admin: up, Oper: up
 Src 10.1.1.1, Dest 10.0.0.9, Instance 43
 Fast Reroute Protection: None
 Path Protection: 0 Common Link(s), 0 Common Node(s)
  Primary lsp path:10.2.0.1  10.2.0.2
                   10.10.0.1 10.10.0.2
                   10.0.0.9
  Protect lsp path:10.0.0.1  10.0.0.2
                   10.0.1.1  10.0.1.2
                   10.0.0.9
  Path Protect Parameters:
   Bandwidth: 100 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
   Metric Type: TE (default)
  InLabel : -
  OutLabel : Ethernet0/0, 17
  RSVP Signalling Info:
   Src 10.1.1.1,  Dst 10.0.0.9,  Tun_Id 500,  Tun_Instance 44
  RSVP Path Info:
   My Address: 10.0.0.1
   Explicit Route: 10.0.0.2  10.0.1.1  10.0.1.2  10.0.0.9
```

```
   Record Route: NONE
   Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  RSVP Resv Info:
   Record Route: NONE
   Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
R1#
```

The following example shows how to shut down the interface to use path protection:

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# interface e1/0
switch(config-if)# shutdown
```

The following example shows that the protection path is used and the primary path is down:

```
switch# show mpls traffic-eng tunnels tunnel-te500

Name: R1_t500 (Tunnel-te500) Destination: 10.0.0.9
 Status:
  Admin: up Oper: up Path: valid Signalling: connected
  path protect option 10, type explicit path3441 (Basis for Protect, path weight 20)
  path option 10, type explicit path344
  path option 20, type explicit path345
  Path Protection: Backup lsp in use.
  path protect option 10, type explicit path3441 (Basis for Protect, path weight 20)
  path protect option 20, type explicit path348

 Config Parameters:
  Bandwidth: 100 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute: enabled LockDown: disabled Loadshare: 100 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
  State: explicit path option 10 is active
  BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet0/0, 17
 RSVP Signalling Info:
   Src 10.1.1.1,  Dst 10.0.0.9,  Tun_Id 500,  Tun_Instance 44
  RSVP Path Info:
   My Address: 10.0.0.1
   Explicit Route: 10.0.0.2 10.0.1.1 10.0.1.2 10.0.0.9
   Record Route: NONE
   Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  RSVP Resv Info:
   Record Route: NONE
   Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
 Shortest Unconstrained Path Info:
  Path Weight: 20 (TE)
  Explicit Route: 10.0.0.1  10.0.0.2  10.0.1.1  10.0.1.2  10.0.0.9
 History:
  Tunnel:
   Time since created: 23 minutes, 28 seconds
   Time since path change: 50 seconds
   Number of LSP IDs (Tun_Instances) used: 44
 Current LSP:
  Uptime: 5 minutes, 24 seconds
  Selection:
 Prior LSP:
  ID: path option 10 [43]
```

```
  Removal Trigger: path error
  Last Error: PCALC:: Explicit path has unknown address, 10.2.0.1
R1#
```

The following example shows that protection is enabled:

```
switch# show mpls traffic-eng tunnels tunnel-te500 protection

R1_t500
 LSP Head, Tunnel-te500, Admin: up, Oper: up
 Src 10.1.1.1, Dest 10.0.0.9, Instance 44
 Fast Reroute Protection: None
 Path Protection: Backup lsp in use.
R1#
```

The following example shows that the interface is up again and the primary path is activated:

```
switch# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet1/0
switch(config-if)# no shutdown
```

The following example shows that path protection has been reestablished and the primary path is being used:

```
switch# show mpls traffic-eng tunnels tunnel-te500

Name: R1_t500 (Tunnel-te500) Destination: 10.0.0.9
 Status:
  Admin: up Oper: up Path: valid Signalling: connected
  path option 10, type explicit path344 (Basis for Setup, path weight 20)
  path option 20, type explicit path345
  Path Protection: 0 Common Link(s), 0 Common Node(s)
  path protect option 10, type explicit path3441 (Basis for Protect, path weight 20)
  path protect option 20, type explicit path348

 Config Parameters:
  Bandwidth: 100 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute: enabled LockDown: disabled Loadshare: 100 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
  State: explicit path option 10 is active
  BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet1/0, 16
 RSVP Signalling Info:
    Src 10.1.1.1, Dst 10.0.0.9, Tun_Id 500, Tun_Instance 52
  RSVP Path Info:
    My Address: 10.2.0.1
    Explicit Route: 10.2.0.2  10.10.0.1  10.10.0.2  10.0.0.9
    Record Route: NONE
    Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  RSVP Resv Info:
    Record Route: NONE
    Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
  Shortest Unconstrained Path Info:
    Path Weight: 20 (TE)
    Explicit Route: 10.0.0.1  10.0.0.2  10.0.1.1  10.0.1.2  10.0.0.9
  History:
    Tunnel:
```

```
    Time since created: 25 minutes, 26 seconds
    Time since path change: 23 seconds
    Number of LSP IDs (Tun_Instances) used: 52
   Current LSP:
   Uptime: 26 seconds
    Selection: reoptimization
   Prior LSP:
    ID: path option 10 [44]
    Removal Trigger: reoptimization completed
switch#
```

The following example shows that Tunnel-te500 is protected and after a failure, the primary LSP is protected:

```
switch# show mpls traffic-eng tunnels tunnel-te500 protection

R1_t500
 LSP Head, Tunnel-te500, Admin: up, Oper: up
 Src 10.1.1.1, Dest 10.0.0.9, Instance 52
 Fast Reroute Protection: None
 Path Protection: 0 Common Link(s), 0 Common Node(s)
  Primary lsp path:10.2.0.1   10.2.0.2
                   10.10.0.1  10.10.0.2
                   10.0.0.9
  Protect lsp path:10.0.0.1   10.0.2
                   10.0.1.1 10.0.1.2
                   10.0.0.9
  Path Protect Parameters:
   Bandwidth: 100 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
   Metric Type: TE (default)
  InLabel : -
  OutLabel : Ethernet0/0, 16
  RSVP Signalling Info:
    Src 10.1.1.1, Dst 10.0.0.9, Tun_Id 500, Tun_Instance 53
   RSVP Path Info:
    My Address: 10.0.0.1
    Explicit Route: 10.0.0.2  10.0.1.1  10.0.1.2  10.0.0.9
    Record Route: NONE
    Tspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
   RSVP Resv Info:
    Record Route: NONE
    Fspec: ave rate=100 kbits, burst=1000 bytes, peak rate=100 kbits
switch#
```

Examples of Enhanced Path Protection

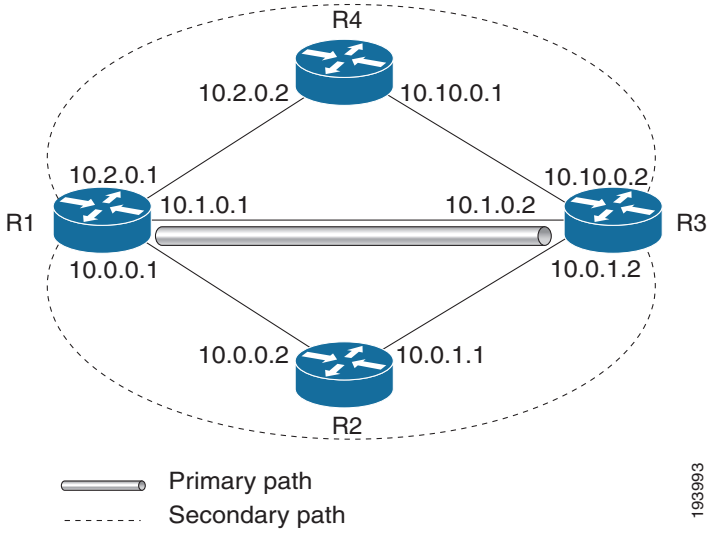
This section includes the following topics:

- Example: Creating a Path Option List, page 17-263
- Example: Assigning a Path Option List to Protect a Primary Path Option, page 17-265
- Example: Configuring Tunnels Before and After Path Protection, page 17-265

Example: Creating a Path Option List

Figure 17-6 shows the network topology for enhanced path protection.

Figure 17-6 Network Topology for Enhanced Path Protection



The following example shows how to configures two explicit paths named **secondary1** and **secondary2**.

```
switch(config)# ip explicit-path name secondary1
switch(config-te-expl-path)# index 1 next 10.0.0.2
Explicit Path name secondary1:
 1: next-address 10.0.0.2

switch(config-te-expl-path)# index 2 next 10.0.1.2
Explicit Path name secondary1:
 1: next-address 10.0.0.2
 2: next-address 10.0.1.2

switch(config-te-expl-path)# ip explicit-path name secondary2
switch(config-te-expl-path)# index 1 next 10.2.0.2
Explicit Path name secondary2:
 1: next-address 10.2.0.2

switch(config-te-expl-path)# index 2 next 10.10.0.2
Explicit Path name secondary2:
 1: next-address 10.2.0.2
 2: next-address 10.10.0.2

switch(config-te-expl-path)# exit
```

The following example shows that a path option list of backup paths is created. You define the path option list by using the explicit paths.

```
switch(config)# mpls traffic-eng
switch(config-te)# path-option list name pathlist-01
switch(cfg-pathoption-list)# path-option 10 explicit name secondary1
path-option 10 explicit name secondary1

switch(cfg-pathoption-list)# path-option 20 explicit name secondary2
path-option 10 explicit name secondary1
path-option 20 explicit name secondary2

switch(cfg-pathoption-list)# exit
```

Example: Assigning a Path Option List to Protect a Primary Path Option

The following example shows that a traffic engineering tunnel is configured:

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# interface tunnel 2
switch(config-if)# tunnel mpls traffic-eng path-option protect 10 list name secondary-list
```

The following example shows that path protection has been configured. Tunnel 2 has path option 10 using path primary1 and protected by secondary-list.

```
switch# show running-config interface tunnel 2

Building configuration...

Current configuration : 296 bytes
!
interface Tunnel2
 ip unnumbered Loopback0
 tunnel mode mpls traffic-eng
 tunnel destination 103.103.103.103
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng path-option 10 explicit name primary1
 tunnel mpls traffic-eng path-option protect 10 list name secondary-list
```

Example: Configuring Tunnels Before and After Path Protection

The following example shows information about the primary (protected) path. The following sample output shows that path protection has been configured:

```
switch# show mpls traffic-eng tunnels tunnel 2

Name: Router_t2 (Tunnel2) Destination: 103.103.103.103
 Status:
  Admin: up Oper: up Path: valid Signalling: connected
  path option 10, type explicit primary1 (Basis for Setup, path weight 10)
  Path Protection: 0 Common Link(s), 0 Common Node(s)
  path protect option 10, type list name secondary-list
  Inuse path-option 10, type explicit secondary1 (Basis for Protect, path weight 20)

 Config Parameters:
  Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute announce: enabled LockDown: disabled Loadshare: 0 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
  State: explicit path option 10 is active
  BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet7/0, implicit-null
 RSVP Signalling Info:
  Src 100.100.100.100, Dst 103.103.103.103, Tun_Id 2, Tun_Instance 11
 RSVP Path Info:
  My Address: 10.1.0.1
  Explicit Route: 10.1.0.2 103.103.103.103
  Record Route: NONE
  Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
 RSVP Resv Info:
  Record Route: NONE
  Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
```

```
 Shortest Unconstrained Path Info:
  Path Weight: 10 (TE)
  Explicit Route: 10.1.0.1 10.1.0.2 103.103.103.103
 History:
  Tunnel:
   Time since created: 24 minutes, 15 seconds
   Time since path change: 23 minutes, 30 seconds
  Number of LSP IDs (Tun_Instances) used: 11
  Current LSP:
   Uptime: 23 minutes, 30 seconds
```

The following shows information about the secondary path. Tunnel 2 is protected.

```
switch# show mpls traffic-eng tunnels tunnel 2 protection

Router_t2
 LSP Head, Tunnel2, Admin: up, Oper: up
 Src 100.100.100.100, Dest 103.103.103.103, Instance 11
 Fast Reroute Protection: None
 Path Protection: 0 Common Link(s), 0 Common Node(s)
  Primary lsp path:10.1.0.1 10.1.0.2
                   103.103.103.103
  Protect lsp path:10.0.0.1 10.0.0.2
                   10.0.1.1 10.0.1.2
                   103.103.103.103
  Path Protect Parameters:
   Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
   Metric Type: TE (default)
  InLabel : -
  OutLabel : Ethernet5/0, 16
  RSVP Signalling Info:
   Src 100.100.100.100, Dst 103.103.103.103, Tun_Id 2, Tun_Instance 20
  RSVP Path Info:
   My Address: 10.0.0.1
   Explicit Route: 10.0.0.2 10.0.1.1 10.0.1.2 103.103.103.103
   Record Route: NONE
   Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
  RSVP Resv Info:
   Record Route: NONE
   Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
```

The following example shows how to shut down the interface to use path protection:

```
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# interface e7/0
switch(config-if)# shutdown
switch(config-if)# end
```

The following example shows that the protection path is used, and the primary path is down:

```
switch# show mpls traffic-eng tunnels tunnel 2

Name: Router_t2 (Tunnel2) Destination: 103.103.103.103
 Status:
  Admin: up Oper: up Path: valid Signalling: connected
  path protect option 10, type list name secondary-list
  Inuse path-option 10, type explicit secondary1 (Basis for Protect, path weight 20)
  path option 10, type explicit primary1
  Path Protection: Backup lsp in use.
  path protect option 10, type list name secondary-list
  Inuse path-option 10, type explicit secondary1 (Basis for Protect, path weight 20)
```

```
 Config Parameters:
  Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute announce: enabled LockDown: disabled Loadshare: 0 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
  State: list path option 10 is active
  BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet5/0, 16
 RSVP Signalling Info:
  Src 100.100.100.100, Dst 103.103.103.103, Tun_Id 2, Tun_Instance 20
 RSVP Path Info:
  My Address: 10.0.0.1
  Explicit Route: 10.0.0.2 10.0.1.1 10.0.1.2 103.103.103.103
  Record Route: NONE
  Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
 RSVP Resv Info:
  Record Route: NONE
  Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
 Shortest Unconstrained Path Info:
  Path Weight: 20 (TE)
  Explicit Route: 10.0.0.1 10.0.0.2 10.0.1.1 10.0.1.2 103.103.103.103

 History:
  Tunnel:
   Time since created: 32 minutes, 27 seconds
   Time since path change: 1 minutes, 7 seconds
   Number of LSP IDs (Tun_Instances) used: 20
  Current LSP:
   Uptime: 8 minutes, 56 seconds
   Selection:
  Prior LSP:
   ID: path option 10 [11]
   Removal Trigger: path error
   Last Error: PCALC:: No addresses to connect 100.100.100.100 to 10.1.0.2
```

The following example shows that protection is enabled.

```
switch# show mpls traffic-eng tunnels tunnel 2 protection

Router_t2
 LSP Head, Tunnel2, Admin: up, Oper: up
 Src 100.100.100.100, Dest 103.103.103.103, Instance 20
 Fast Reroute Protection: None
 Path Protection: Backup lsp in use.
```

The following example shows that the interface is up again and the primary path is activated.

```
switch# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.
switch(config-if)# interface ethernet7/0
switch(config-if)# no shutdown
switch(config-if)# end
```

The following example shows that path protection has been reestablished and the primary path is being used:

```
switch# show mpls traffic-eng tunnels tunnel 2

Name: Router_t2 (Tunnel2) Destination: 103.103.103.103
 Status:
```

```
  Admin: up Oper: up Path: valid Signalling: connected
  path option 10, type explicit primary1 (Basis for Setup, path weight 10)
  Path Protection: 0 Common Link(s), 0 Common Node(s)
  path protect option 10, type list name secondary-list
  Inuse path-option 10, type explicit secondary1 (Basis for Protect, path weight 20)

 Config Parameters:
  Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute announce: enabled LockDown: disabled Loadshare: 0 bw-based
  auto-bw: disabled
 Active Path Option Parameters:
  State: explicit path option 10 is active
  BandwidthOverride: disabled LockDown: disabled Verbatim: disabled

 InLabel : -
 OutLabel : Ethernet7/0, implicit-null
 RSVP Signalling Info:
    Src 100.100.100.100, Dst 103.103.103.103, Tun_Id 2, Tun_Instance 39
  RSVP Path Info:
    My Address: 10.1.0.1
    Explicit Route: 10.1.0.2 103.103.103.103
    Record Route: NONE
    Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
  RSVP Resv Info:
    Record Route: NONE
    Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
  Shortest Unconstrained Path Info:
    Path Weight: 10 (TE)
    Explicit Route: 10.1.0.1 10.1.0.2 103.103.103.103
  History:
    Tunnel:
    Time since created: 40 minutes, 59 seconds
    Time since path change: 1 minutes, 24 seconds
    Number of LSP IDs (Tun_Instances) used: 39
  Current LSP:
    Uptime: 1 minutes, 27 seconds
    Selection: reoptimization
  Prior LSP:
    ID: path option 10 [20]
    Removal Trigger: reoptimization completed

switch# show mpls traffic-eng tunnels tunnel 2 protection

Router_t2
 LSP Head, Tunnel2, Admin: up, Oper: up
 Src 100.100.100.100, Dest 103.103.103.103, Instance 39
 Fast Reroute Protection: None
 Path Protection: 0 Common Link(s), 0 Common Node(s)
  Primary lsp path:10.1.0.1 10.1.0.2
                   103.103.103.103
  Protect lsp path:10.0.0.1 10.0.0.2
                   10.0.1.1 10.0.1.2
                   103.103.103.103
  Path Protect Parameters:
    Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
    Metric Type: TE (default)
  InLabel : -
  OutLabel : Ethernet5/0, 17
  RSVP Signalling Info:
    Src 100.100.100.100, Dst 103.103.103.103, Tun_Id 2, Tun_Instance 40
  RSVP Path Info:
    My Address: 10.0.0.1
    Explicit Route: 10.0.0.2 10.0.1.1 10.0.1.2 103.103.103.103
```

```
   Record Route: NONE
   Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
  RSVP Resv Info:
   Record Route: NONE
   Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
```

The following example shows the contents of the RSVP high availability read and write databases used in TE.

```
switch# show ip rsvp high-availability database lsp-head

LSP_HEAD WRITE DB
 Tun ID: 2
 Header:
  State: Checkpointed Action: Modify
  Seq #: 17 Flags: 0x0
 Data:
  lsp_id: 39, bandwidth: 0, thead_flags: 0x1, popt: 10
  feature flags: none
  output_if_num: 31, output_nhop: 10.1.0.2
 RRR path setup info
   Destination: 103.103.103.103, Id: 103.103.103.103 Router Node (ospf) flag:0x0
   IGP: ospf, IGP area: 0, Number of hops: 3, metric: 10
   Hop 0: 10.1.0.1, Id: 100.100.100.100 Router Node (ospf), flag:0x0
   Hop 1: 10.1.0.2, Id: 103.103.103.103 Router Node (ospf), flag:0x0
   Hop 2: 103.103.103.103, Id: 103.103.103.103 Router Node (ospf), flag:0x0

LSP_HEAD READ DB
```

Additional References for MPLS TE Path Protection

The following sections provide references related to the MPLS TE path protection feature.

Related Documents

Related Topic	Document Title
MPLS TE commands	*Cisco NX-OS Multiprotocol Label Switching Command Reference*
RSVP	*Cisco NX-OS Quality of Service Commands*
IS-IS	*Cisco NX-OS Unicast Routing Protocols Command Reference* *Configuring a Basic IS-IS Network*
OSPF	*Cisco NX-OS Unicast Routing Protocols Command Reference* *Configuring OSPF*
ISSU	*ISSU MPLS Clients* *Cisco NX-OS High Availability and Redundancy Guide*
NSF/SSO	*Cisco NX-OS High Availability and Redundancy Guide*

MIBs

MIB	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for MPLS TE Path Protection

Table 17-1 lists the release history for this feature.

Table 17-1 Feature History for MPLS TE Path Protection

Feature Name	Releases	Feature Information
Path protection	5.2(1)	This feature was introduced.

Configuring MPLS TE Fast Reroute Link and Node Protection

This chapter describes how to configure Multiprotocol Label Switching (MPLS) traffic engineering (TE) fast reroute link and node protection on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS TE Fast Reroute Link and Node Protection

Fast reroute link and node protection provides link protection (backup tunnels that bypass only a single link of the label switched path [LSP]), node protection (backup tunnels that bypass next-hop nodes along LSPs), and the following fast reroute (FRR) features:

- Backup tunnel support
- Backup bandwidth protection
- Bidirectional Forwarding Detection (BFD)

This section includes the following topics:

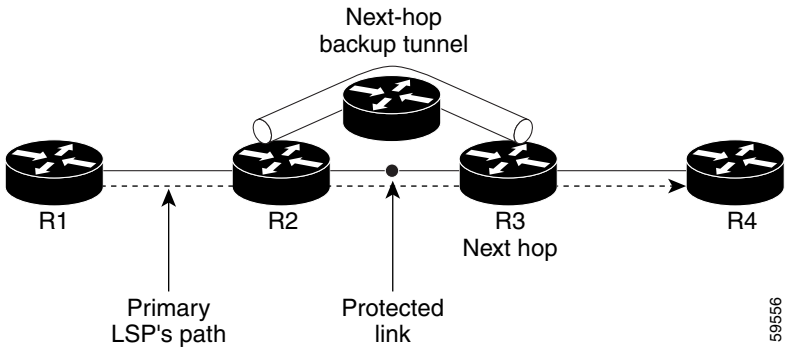
Fast Reroute

Fast Reroute (FRR) is a mechanism for protecting MPLS TE LSPs from link and node failures by locally repairing the LSPs at the point of failure, allowing data to continue to flow on them while their headend routers try to establish new end-to-end LSPs to replace them. FRR locally repairs the protected LSPs by rerouting them over backup tunnels that bypass failed links or node.

Link Protection

Backup tunnels that bypass only a single link of the LSP's path provide link protection. They protect LSPs if a link along their path fails by rerouting the LSP's traffic to the next hop (bypassing the failed link). These tunnels are referred to as next-hop (NHOP) backup tunnels because they terminate at the LSP's next hop beyond the point of failure. Figure 18-1 shows an NHOP backup tunnel.

Figure 18-1 NHOP Backup Tunnel

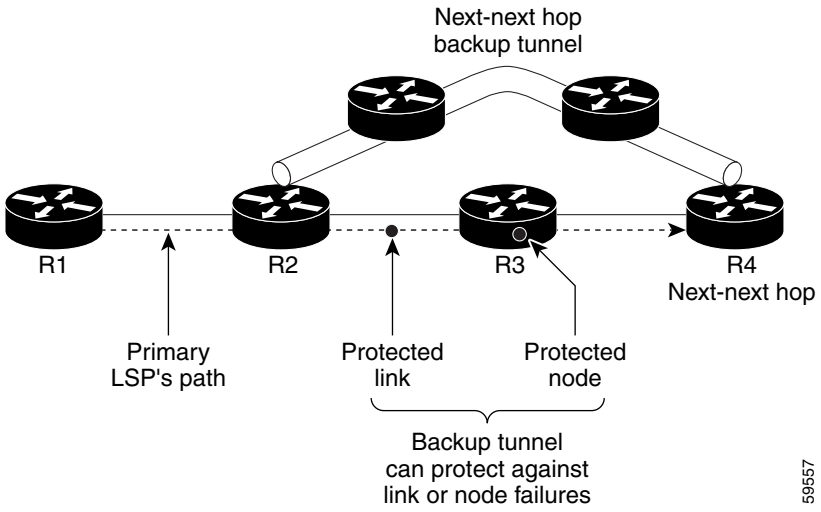


Node Protection

FRR provides node protection for LSPs. Backup tunnels that bypass next-hop nodes along LSP paths are called next-next-hop (NNHOP) backup tunnels because they terminate at the node following the next-hop node of the LSP paths, bypassing the next-hop node. They protect LSPs if a node along their path fails by enabling the node upstream of the failure to reroute the LSPs and their traffic around the failed node to the next-next hop. FRR supports the use of BFD to accelerate the detection of node failures. NNHOP backup tunnels also provide protection from link failures, because they bypass the failed link and the node.

Figure 18-2 shows an NNHOP backup tunnel.

Figure 18-2 NNHOP Backup Tunnel



If an LSP is using a backup tunnel and something changes so that the LSP is no longer appropriate for the backup tunnel, the LSP is torn down. Such changes are the following:

- The backup bandwidth of the backup tunnel is reduced.
- The backup bandwidth type of the backup tunnel is changed to a type that is incompatible with the primary LSP.
- The primary LSP is modified so that FRR is disabled. (The **no fast-reroute** command is entered.)

Bandwidth Protection

NHOP and NNHOP backup tunnels can be used to provide bandwidth protection for rerouted LSPs, which is referred to as backup bandwidth. You can associate backup bandwidth with NHOP or NNHOP backup tunnels to inform the router of the amount of backup bandwidth a particular backup tunnel can protect. When a router maps LSPs to backup tunnels, bandwidth protection ensures that an LSP uses a given backup tunnel only if there is sufficient backup bandwidth. The router selects which LSPs use which backup tunnels in order to provide maximum bandwidth protection. The router determines the best way to map LSPs onto backup tunnels in order to maximize the number of LSPs that can be protected. For information about mapping tunnels and assigning backup bandwidth, see the "Backup Tunnel Selection Procedure" section on page 18-279.

LSPs that have the bandwidth protection desired bit set have a higher right to select backup tunnels that provide bandwidth protection; that is, those LSPs can preempt other LSPs that do not have that bit set. For more information, see the "Prioritizing Which LSPs Obtain Backup Tunnels with Bandwidth Protection" section on page 18-276.

Features of Fast Reroute Link and Node Protection

Fast reroute link and node protection has the following features:

Backup Tunnel Support

This section includes the following topics:

Backup Tunnels Can Terminate at the Next-Next Hop to Support FRR

Backup tunnels that terminate at the next-next hop protect both the downstream link and node, which provides protection for link and node failures. For more detailed information, see the "Node Protection" section on page 18-274.

Multiple Backup Tunnels Can Protect the Same Interface

There is no limit (except memory limitations) to the number of backup tunnels that can protect a given interface. In many topologies, support for node protection requires supporting multiple backup tunnels per protected interface. These backup tunnels can terminate at the same destination or at different destinations. For a given protected interface, you can configure multiple NHOP or NNHOP backup tunnels, which allow redundancy and load balancing.

In addition to being required for node protection, the protection of an interface by multiple backup tunnels provides the following benefits:

- Redundancy—If one backup tunnel is down, other backup tunnels protect LSPs.
- Increased backup capacity—If the protected interface is a high-capacity link and no single backup path exists with an equal capacity, multiple backup tunnels can protect that one high-capacity link. The LSPs using this link fails over to different backup tunnels, allowing all of the LSPs to have adequate bandwidth protection during the failure (rerouting). If bandwidth protection is not desired, the router spreads LSPs across all available backup tunnels (that is, there is load balancing across backup tunnels). For more details, see the "Backup Tunnel Selection Procedure" section on page 18-279.

Examples are shown in the "Backup Tunnels Terminating at Different Destinations" section on page 18-277 and the "Backup Tunnels Terminating at the Same Destination" section on page 18-278.

Backup Tunnels Provide Scalability

A backup tunnel can protect multiple LSPs and multiple interfaces. This feature is called many-to-one (N:1) protection. An example of N:1 protection is when one backup tunnel protects 5000 LSPs, where each router along the backup path maintains one additional tunnel.

One-to-one protection is when a separate backup tunnel must be used for each LSP that needs protection. N:1 protection has significant scalability advantages over one-to-one (1:1) protection. An example of 1:1 protection is when 5000 backup tunnels protect 5000 LSPs, where each router along the backup path must maintain the state for an additional 5000 tunnels.

Backup Bandwidth Protection

Backup bandwidth protection allows you to give LSPs that carry certain kinds of data (such as voice) priority for using backup tunnels. Backup bandwidth protection has the following capabilities:

- Bandwidth Protection on Backup Tunnels, page 18-276
- Bandwidth Pool Specifications for Backup Tunnels, page 18-276
- Semidynamic Backup Tunnel Paths, page 18-276
- Prioritizing Which LSPs Obtain Backup Tunnels with Bandwidth Protection, page 18-276

Bandwidth Protection on Backup Tunnels

Rerouted LSPs not only have their packets delivered during a failure, but the quality of service can also be maintained.

Bandwidth Pool Specifications for Backup Tunnels

You can restrict the types of LSPs that can use a given backup tunnel. Backup tunnels can be restricted so that only LSPs that use global-pool bandwidth can use them. This feature allows different backup tunnels to be used for voice and data. Example: The backup tunnel used for voice could provide bandwidth protection, and the backup tunnel used for data could not provide bandwidth protection.

Semidynamic Backup Tunnel Paths

The path of a backup tunnel can be configured to be determined dynamically, which can be done by using the IP explicit address exclusion feature. If you use this feature, semidynamic NHOP backup tunnel paths can be specified simply by excluding the protected link; semidynamic NNHOP backup tunnel paths can be configured simply by excluding the protected node.

Prioritizing Which LSPs Obtain Backup Tunnels with Bandwidth Protection

In case there are not enough NHOP or NNHOP backup tunnels or they do not have enough backup bandwidth to protect all LSPs, you can give an LSP priority in obtaining backup tunnels with bandwidth protection. This feature is especially useful if you want to give LSPs that carry voice a higher priority than those LSPs that carry data.

To activate this feature, enter the **tunnel mpls traffic-eng fast-reroute bw-protect** command to set the bandwidth protection desired bit. See the "Enabling Fast Reroute on LSPs" section on page 18-287.

The LSPs do not necessarily receive bandwidth protection. They have a higher chance of receiving bandwidth protection if they need it.

LSPs that do not have the bandwidth protection bit set can be demoted. Demotion is when one or more LSPs are removed from their assigned backup tunnel to provide backup to an LSP that has its bandwidth protection bit set. Demotion occurs only when there is a scarcity of backup bandwidth.

When an LSP is demoted, it becomes unprotected (that is, it no longer has a backup tunnel). During the next periodic promotion cycle, an attempt is made to find the best possible backup tunnels for all LSPs that do not currently have protection, including the LSP that was demoted. The LSP may get protection at the same level or a lower level, or it may get no protection.

For information about how routers determine which LSPs to demote, see the "Backup Protection Preemption Algorithms" section on page 18-282.

Bidirectional Forwarding Detection

The Bidirectional Forwarding Detection (BFD) triggered fast reroute feature enables you to obtain link and node protection by using the BFD protocol to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators.

BFD for MPLS TE fast reroute is enabled as soon as fast reroute is enabled on a tunnel interface.

Fast Reroute Operation

This section includes the following topics:

Fast Reroute Activation

Two mechanisms cause routers to switch LSPs onto their backup tunnels:

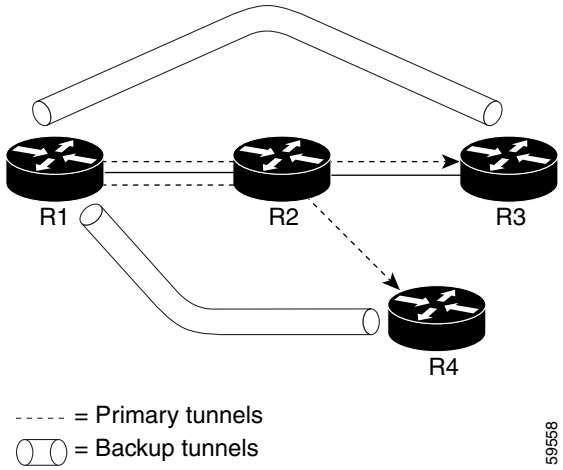
- Interface down notification
- BFD neighbor down notification

When a router's link or neighboring node fails, the router often detects this failure by an interface down notification. When a router notices that an interface has gone down, it switches LSPs going out that interface onto their respective backup tunnels (if any).

Backup Tunnels Terminating at Different Destinations

Figure 18-3 shows an interface that has multiple backup tunnels that terminate at different destinations and demonstrates why, in many topologies, support for node protection requires supporting multiple backup tunnels per protected interface.

Figure 18-3 Backup Tunnels that Terminate at Different Destinations



In this figure, a single interface on R1 requires multiple backup tunnels. LSPs traverse the following routes:

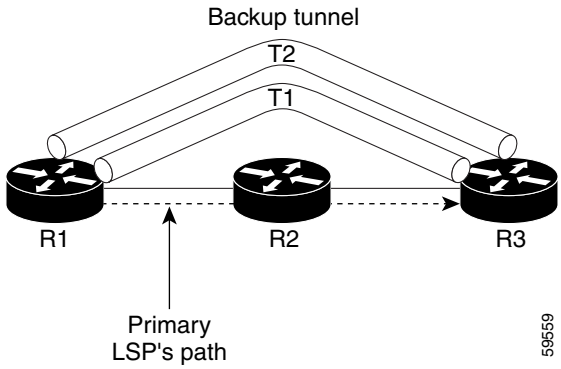
- R1, R2, R3
- R1, R2, R4

To provide protection if node R2 fails, two NNHOP backup tunnels are required: one terminating at R3 and one terminating at R4.

Backup Tunnels Terminating at the Same Destination

Figure 18-4 shows how backup tunnels that terminate at the same location can be used for redundancy and load balancing. Redundancy and load balancing work for both NHOP and NNHOP backup tunnels.

Figure 18-4 Backup Tunnels that Terminate at the Same Destination



In this figure, there are three routers: R1, R2, and R3. At R1, two NNHOP backup tunnels (T1 and T2) go from R1 to R3 without traversing R2.

Redundancy—If R2 fails or the link from R1 to R2 fails, either backup tunnel can be used. If one backup tunnel is down, the other can be used. LSPs are assigned to backup tunnels when the LSPs are first established before a failure occurs.

Load balancing—If neither backup tunnel has enough bandwidth to back up all LSPs, both tunnels can be used. Some LSPs use one backup tunnel, while other LSPs use the other backup tunnel. The router decides the best way to fit the LSPs onto the backup tunnels.

Backup Tunnel Selection Procedure

When an LSP is signaled, each node along the LSP path that provides FRR protection for the LSP selects a backup tunnel for the LSP to use if either of the following events occurs:

- The link to the next hop fails.
- The next hop fails.

By having the node select the backup tunnel for an LSP before a failure occurs, the LSP can be rerouted onto the backup tunnel quickly if there is a failure.

For an LSP to be mapped to a backup tunnel, all of the following conditions must exist:

- The LSP is protected by FRR; that is, the LSP is configured with the **fast-reroute** command.
- The backup tunnel is up.
- The backup tunnel is configured to have an IP address which is typically a loopback address.
- The backup tunnel is configured to protect this LSP's outgoing interface; that is, the interface is configured with the **mpls traffic-eng backup-path** command.
- The backup tunnel does not traverse the LSP's protected interface.
- The backup tunnel terminates at the LSP's NHOP or NNHOP. If it is an NNHOP tunnel, it does not traverse the LSP's NHOP.
- The bandwidth protection requirements and constraints, if any, for the LSP and backup tunnel are met. For information about bandwidth protection considerations, see the "Bandwidth Protection" section on page 18-279.

Bandwidth Protection

A backup tunnel can be configured to protect two types of backup bandwidth:

- Limited backup bandwidth—A backup tunnel provides bandwidth protection. The sum of the bandwidth of all LSPs using this backup tunnel cannot exceed the backup tunnel's backup bandwidth. When you assign LSPs to this type of backup tunnel, sufficient backup bandwidth must exist.
- Unlimited backup bandwidth—The backup tunnel does not provide any bandwidth protection (that is, best-effort protection exists). There is no limit to the amount of bandwidth used by the LSPs that are mapped to this backup tunnel. LSPs that allocate zero bandwidth can use only backup tunnels that have unlimited backup bandwidth.

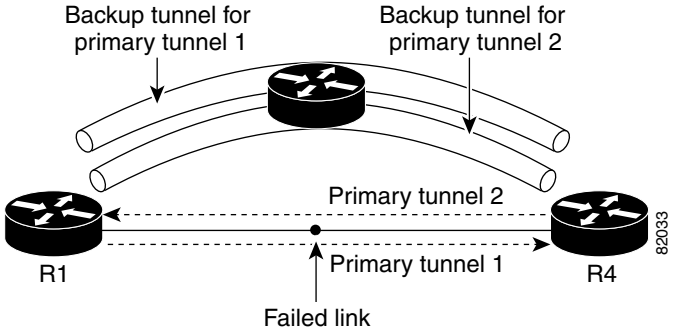
Load Balancing on Limited-Bandwidth Backup Tunnels

There may be more than one backup tunnel that has sufficient backup bandwidth to protect a given LSP. In this case, the router chooses the one that has the least amount of backup bandwidth available. This algorithm limits fragmentation, maintaining the largest amount of backup bandwidth available.

Specifying limited backup bandwidth does not guarantee bandwidth protection if there is a link or node failure. For example, the set of NHOP and NNHOP backup tunnels that gets triggered when an interface fails may all share some link on the network topology, and this link may not have sufficient bandwidth to support all LSPs using this set of backup tunnels.

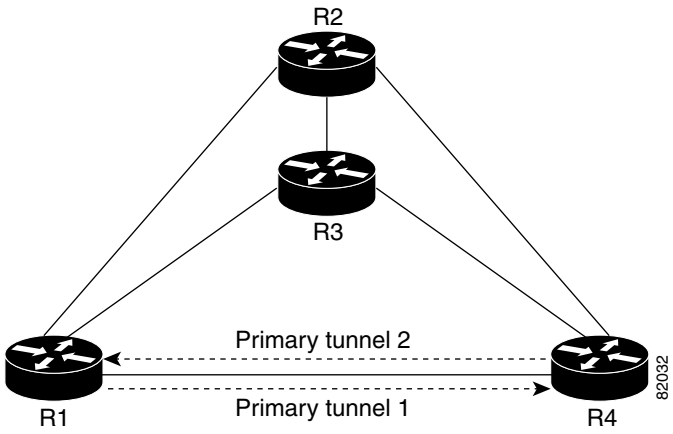
In Figure 18-5, both backup tunnels traverse the same links and hop. When the link between routers R1 and R4 fails, backup tunnels for primary tunnel 1 and primary tunnel 2 are triggered simultaneously. The two backup tunnels may share a link in the network.

Figure 18-5 Backup Tunnels Share a Link



In Figure 18-6, the backup tunnel for primary tunnel 1 may traverse routers R1-R2-R3-R4, and the backup tunnel for primary tunnel 2 might traverse routers R4-R2-R3-R1. In this case, the link R2-R3 might get overloaded if R1-R4 fails.

Figure 18-6 Overloaded Link



Load Balancing on Unlimited-Bandwidth Backup Tunnels

More than one backup tunnel, each having unlimited backup bandwidth, can protect a given interface. In this case, when choosing a backup tunnel for a given LSP, the router chooses the backup tunnel that has the least amount of backup bandwidth in use. This algorithm evenly distributes the LSPs across backup tunnels based on an LSP's bandwidth. If an LSP is requesting zero bandwidth, the router chooses the backup tunnel that is protecting the fewest LSPs.

Pool Type and Backup Tunnels

By default, a backup tunnel provides protection for LSPs that allocate from any global pool. However, a backup tunnel can be configured to protect only LSPs that use global-pool bandwidth.

Tunnel Selection Priorities

This section includes the following topics:

NHOP Versus NNHOP Backup Tunnels

More than one backup tunnel can protect a given LSP, where one backup tunnel terminates at the LSP's NNHOP, and the other terminates at the LSP's NHOP. In this case, the router chooses the backup tunnel that terminates at the NNHOP (FRR prefers NNHOP over NHOP backup tunnels).

Table 18-1 lists the tunnel selection priorities. The first choice is an NNHOP backup tunnel that acquires its bandwidth from a global pool and has limited bandwidth. If there is no such backup tunnel, the next choice (2) is a next-next hop backup tunnel that acquires a limited amount of bandwidth from any pool. The preferences go from 1 (best) to 8 (worst), where choice 3 is for an NNHOP backup tunnel with an unlimited amount of global-pool bandwidth.

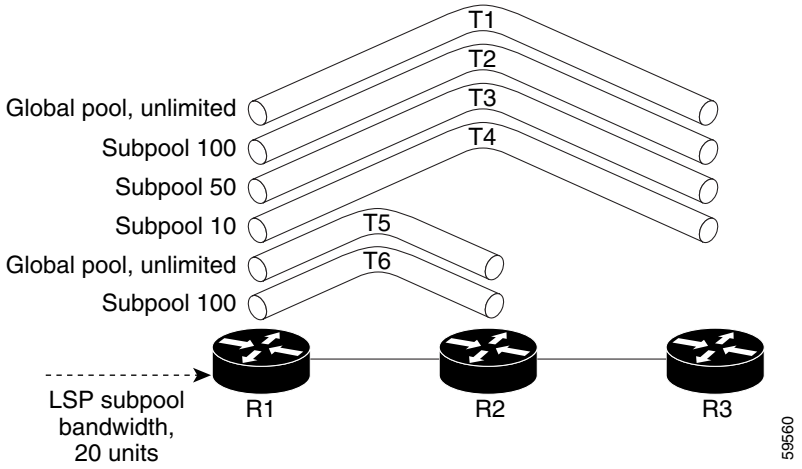
Table 18-1 Tunnel Selection Priorities

Preference	Backup Tunnel Destination	Bandwidth Pool	Bandwidth Amount
1 (Best)	NNHOP	Global pool	Limited
2	NNHOP	Any	Limited
3	NNHOP	Global pool	Unlimited
4	NNHOP	Any	Unlimited
5	NHOP	Global pool	Limited
6	NHOP	Any	Limited
7	NHOP	Global pool	Unlimited
8 (Worst)	NHOP	Any	Unlimited

Figure 18-7 shows an example of the backup tunnel selection procedure based on the designated amount of global pool bandwidth currently available.

Note If NHOP and NNHOP backup tunnels do not have sufficient backup bandwidth, no consideration is given to the type of data that the LSP is carrying. For example, a voice LSP may not be protected unless it is signaled before a data LSP. To prioritize the backup tunnel usage, see the "Backup Protection Preemption Algorithms" section on page 18-282.

Figure 18-7 Choosing from Among Multiple Backup Tunnels



In this example, an LSP requires 20 units (kilobits per second) of global pool backup bandwidth. The best backup tunnel is selected as follows:

1. Backup tunnels T1 is considered first because it terminates at the NNHOP.
2. Tunnel T1 is chosen because it protects LSPs using global pool bandwidth.
3. Tunnels T5 is not considered because it terminates at an NHOP, and therefore is less desirable than T1, which terminates at an NNHOP.

Promotion

After a backup tunnel has been chosen for an LSP, conditions may change that will cause you to reevaluate this choice. This reevaluation, if successful, is called promotion. Such conditions are as follows:

1. A new backup tunnel comes up.
2. The currently chosen backup tunnel for this LSP goes down.
3. A backup tunnel's available backup bandwidth increases. For example, an LSP protected by the tunnel has been reoptimized by the headend to use another path.

For cases 1 and 2, the LSP's backup tunnel is evaluated immediately. Case 3 is addressed by periodically reevaluating LSP-to-backup tunnel mappings. By default, background reevaluation is performed every 5 minutes. You can configure this interval with the **fast-reroute timers** command.

Backup Protection Preemption Algorithms

When you set the bandwidth protection desired bit for an LSP, the LSP has a higher right to select backup tunnels that provide bandwidth protection and it can preempt other LSPs that do not have that bit set.

If there is insufficient backup bandwidth on NNHOP backup tunnels but not on NHOP backup tunnels, the bandwidth-protected LSP does not preempt NNHOP LSPs; it uses NHOP protection.

If multiple LSPs are using a given backup tunnel and one or more must be demoted to provide bandwidth, there are two user-configurable methods (algorithms) that the router can use to determine which LSPs are demoted:

- Minimize amount of bandwidth that is wasted.
- Minimize the number of LSPs that are demoted.

For example, if you need ten units of backup bandwidth on a backup tunnel, you can demote one of the following:

- A single LSP using 100 units of bandwidth—Makes available more bandwidth than needed but results in wasted bandwidth.
- Ten LSPs, each using one unit of bandwidth—Results in no wasted bandwidth but affects more LSPs.

The default algorithm is to minimize the number of LSPs that are demoted. To change the algorithm to minimize the amount of bandwidth that is wasted, enter the **fast-reroute backup-prot-preemption optimize-bw** command.

Bandwidth Protection Considerations

Bandwidth protection can be ensured in many ways. Table 18-2 describes the advantages and disadvantages of three methods.

Table 18-2 Bandwidth Protection Methods

Method	Advantages	Disadvantages
Reserve bandwidth for backup tunnels explicitly.	It is simple.	It is a challenge to allow bandwidth sharing of backup tunnels protecting against independent failures.
Use backup tunnels that are signaled with zero bandwidth.	It provides a way to share bandwidth used for protection against independent failures, so it ensures more economical bandwidth usage.	It may be complicated to determine the proper placement of zero bandwidth tunnels.
Backup bandwidth protection.	It ensures bandwidth protection for voice traffic.	An LSP that does not have backup bandwidth protection can be demoted at any time if there is not enough backup bandwidth and an LSP that has backup bandwidth protection needs bandwidth.

Cisco implementation of FRR does not mandate a particular approach, and it provides the flexibility to use any of the above approaches. However, given a range of configuration choices, be sure that the choices are constant with a particular bandwidth protection strategy.

This section includes the following topics:

- Using Backup Tunnels with Explicitly Signaled Bandwidth, page 18-283
- Protected Bandwidth Pools and the Bandwidth Pool from Which the Backup Tunnel Reserves Its Bandwidth, page 18-284
- Using Backup Tunnels Signaled with Zero Bandwidth, page 18-284
- Signaled Bandwidth Versus Backup Bandwidth, page 18-285

Using Backup Tunnels with Explicitly Signaled Bandwidth

Two bandwidth parameters must be set for a backup tunnel:

- Actual signaled bandwidth

- Backup bandwidth

To signal bandwidth requirements of a backup tunnel, configure the bandwidth of the backup tunnel by using the **bandwidth** command.

To configure the backup bandwidth of the backup tunnel, use the **backup-bw** command.

The signaled bandwidth is used by the LSRs on the path of the backup tunnel to perform admission control and do appropriate bandwidth accounting.

The backup bandwidth is used by the point of local repair (PLR) (the headend of the backup tunnel) to decide how much primary traffic can be rerouted to this backup tunnel if there is a failure.

Both parameters need to be set to ensure proper operation. The numerical value of the signaled bandwidth and the backup bandwidth should be the same.

Protected Bandwidth Pools and the Bandwidth Pool from Which the Backup Tunnel Reserves Its Bandwidth

The **bandwidth** command allows you to configure the following:

- Amount of bandwidth that a backup tunnel reserves
- The DS-TE bandwidth pool from which the bandwidth needs to be reserved

Note Only one pool can be selected. The backup tunnel can explicitly reserve bandwidth from the global pool.

The **backup-bw** command allows you to specify the bandwidth pool to which the traffic must belong for the traffic to use this backup tunnel. Multiple pools are allowed.

There is no direct correspondence between the bandwidth pool that is protected and the bandwidth pool from which the bandwidth of the backup tunnel draws its bandwidth.

Bandwidth protection for 10 Kbps of subpool traffic on a given link can be achieved by configuring any of the following command combinations:

- **bandwidth sub-pool 10**

  backup-bw sub-pool 10
- **bandwidth global-pool 10**

  backup-bw sub-pool 10 global-pool unlimited
- **bandwidth global-pool 40**

  backup-bw sub-pool 10 global-pool 30

Using Backup Tunnels Signaled with Zero Bandwidth

You might use backup tunnels with zero signaled bandwidth, even when bandwidth protection is required. It might seem that if no bandwidth is explicitly reserved, no bandwidth guarantees can be provided. However, that is not necessarily true, because in the following situation, only link protection is desired and bandwidth protection is desired only for subpool traffic.

For each protected link AB with a maximum reservable subpool value of *n*, there might be a path from node A to node B where the difference between the maximum reservable global and the maximum reservable subpool is at least the value of *n*. If it is possible to find such paths for each link in the network, you can establish all the backup tunnels along such paths without any bandwidth reservations. If there is a single link failure, only one backup tunnel uses any link on its path. Because that path has at least *n* available bandwidth (in the global pool), assuming that marking and scheduling is configured to classify the subpool traffic into a priority queue, the subpool bandwidth is guaranteed.

This approach allows sharing of the global pool bandwidth between backup tunnels that protect independent link failures. The backup tunnels are expected to be used for only a short period of time after a failure (until the headends of affected LSPs reroute those LSPs to other paths with available subpool bandwidth). The probability of multiple unrelated link failures is very small (in the absence of node failures, which result in multiple link failures). Therefore, you can assume that link failures are in practice independent with high probability. This independent failure assumption in combination with backup tunnels signaled without explicit bandwidth reservation enables efficient bandwidth sharing that yields substantial bandwidth savings.

Backup tunnels that protect the subpool traffic do not draw bandwidth from any pool. Primary traffic that uses the global pool can use the entire global pool, and primary traffic that uses the subpool can use the entire subpool. Yet, subpool traffic has a complete bandwidth guarantee if there is a single link failure.

A similar approach can be used for node protection. However, the decision of where to put the backup tunnels is more complicated because node failures effectively result in the simultaneous failure of several links. Therefore, the backup tunnels that protect traffic traversing all affected links cannot be computed independently of each other. The backup tunnels that protect groups of links corresponding to different failures can still be computed independently of each other, which results in similar bandwidth savings.

Signaled Bandwidth Versus Backup Bandwidth

Backup bandwidth is used locally (by the router that is the headend of the backup tunnel) to determine which, and how many, primary LSPs can be rerouted on a particular backup tunnel. The router ensures that the combined bandwidth requirement of these LSPs does not exceed the backup bandwidth.

Therefore, even when the backup tunnel is signaled with zero bandwidth, you must configure the backup bandwidth with the value corresponding to the actual bandwidth requirement of the traffic protected by this backup tunnel. Unlike the case when bandwidth requirements of the backup tunnels are explicitly signaled, the value of the signaled bandwidth (which is zero) is not the same value as the backup bandwidth.

Licensing Requirements for MPLS TE Fast Reroute Link and Node Protection

Product	License Requirement
Cisco NX-OS	Fast reroute link and node protection requires an MPLS license. For a complete explanation of the NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS TE Fast Reroute Link and Node Protection

Fast reroute link and node protection feature has the following prerequisites:

- The MPLS TE feature must be enabled. MPLS TE can be enabled or disabled by the [**no**] **feature mpls traffic-eng** command.

- Before configuring FRR link and node protection, you must complete the following tasks but you do not have to already have configured MPLS TE tunnels:
  - Enable MPLS TE on all relevant routers and interfaces

Guidelines and Limitations for MPLS TE Fast Reroute Link and Node Protection

Fast reroute link and node protection has the following guidelines and limitations:

- Interfaces must use MPLS global label allocation. Labels are allocated from the label table that is unique per VDC.
- Backup tunnel headend and tailend routers must implement FRR as described in draft-pan-rsvp-fastreroute-00.txt.
- Backup tunnels are not protected. If an LSP is actively using a backup tunnel and the backup tunnel fails, the LSP is torn down.
- LSPs that are actively using backup tunnels are not considered for promotion. If an LSP is actively using a backup tunnel and a better backup tunnel becomes available, the active LSP is not switched to the better backup tunnel.
- You cannot enable FRR Hellos on a router that also has Resource Reservation Protocol (RSVP) Graceful Restart enabled.
- You cannot enable primary one-hop autotunnels, backup autotunnels, or autotunnel mesh groups on a router that is also configured with stateful switchover (SSO) redundancy. This restriction does not prevent an MPLS TE tunnel that is automatically configured by TE autotunnel from being successfully recovered by any midpoint router along the LSP's path if the router experiences an SSO switchover.
- MPLS TE LSPs that are fast reroutable cannot be successfully recovered if the LSPs are FRR active and the Point of Local Repair (PLR) router experiences an SSO.
- When SSO (stateful switchover) occurs on a router, the switchover process must complete before FRR (fast reroute) can complete successfully. In a testing environment, allow approximately 2 minutes for TE SSO recovery to complete before manually triggering FRR. To check the TE SSO status, use the **show ip rsvp high-availability summary** command. Note the status of the HA state field as follows:
  - When SSO is in the process of completing, this field displays as Recovering.
  - When the SSO process has completed, this field displays as Active.

Configuring MPLS TE Fast Reroute Link and Node Protection

This section includes the following topics:

Enabling Fast Reroute on LSPs

LSPs can use backup tunnels only if they have been configured as fast reroutable. You can enable fast reroute on LSPs by entering commands at the headend of each LSP.

Note This procedure is required for configuring fast reroute link and node protection.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **fast-reroute** [**bw-protect**]

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 1000` `switch(config-if-te)#`	Enters interface configuration mode for the specified tunnel.
Step 3	**fast-reroute** [**bw-protect**] **Example:** `switch(config-if-te)# fast-reroute bw-protect`	Enables an MPLS TE tunnel to use an established backup tunnel if there is a link or node failure.

Creating a Backup Tunnel to the Next Hop or to the Next-Next Hop

Creating a backup tunnel is basically no different from creating any other tunnel. To create a backup tunnel to the next hop or to the next-next hop, enter commands on the node that will be the headend of the backup tunnel (the node whose downstream link or the node might fail). The node on which you enter these commands must be a supported platform.

Note This procedure is required for configuring fast reroute link and node protection.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng configuration**
3. **explicit-path** {**identifier** *id* | **name** *name*}
4. **exclude-address** *ip-address*
5. **exit**
6. **interface tunnel-te** *number*
7. **ip unnumbered** *interface-type interface-number*
8. **destination** *ip-address*
9. **path-option** [**protect**] *preference-number* {**dynamic** | **explicit** {**identifier** *id* | **name** *name*} [**verbatim**]} [**lockdown**] [**bandwidth** *kbps*] [**attributes** *listname*]

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls traffic-eng configuration** **Example:** `switch(config)# mpls traffic-eng configuration` `switch(config-te)#`	Enters MPLS TE configuration mode.
Step 3	**explicit-path [identifier** *id* **| name** *name***]** **Example:** `switch(config-te)# explicit-path name avoid-protected-link`	Enters the command mode for IP explicit paths and creates the specified path. This command allows you to enter explicit path command mode.

	Command	Purpose
Step 4	**exclude-address** *ip-address* **Example:** `switch(config-ip-expl-path)# exclude-address 3.3.3.3`	For link protection, specify the IP address of the link to be protected. For node protection, specify the router ID of the node to be protected. **Note** Backup tunnel paths can be dynamic or explicit and they do not have to use exclude-address. Because backup tunnels must avoid the protected link or node, it is convenient to use the **exclude-address** command. **Note** When using the **exclude-address** command to specify the path for a backup tunnel, you must exclude an interface IP address to avoid a link (for creating an NHOP backup tunnel), or a router ID address to avoid a node (for creating an NNHOP backup tunnel).
Step 5	**exit** **Example:** `switch(config-te-expl-path)# exit` `switch(config-if)#`	Exits explicit path command mode.
Step 6	**interface tunnel-te** *number* **Example:** `switch(config-if)# interface tunnel-te 1` `switch(config-if-te)#`	Creates a new tunnel interface and enters interface configuration mode.
Step 7	**ip unnumbered** *interface-type interface-number* **Example:** `switch(config-if-te)# ip unnumbered loopback 0`	Gives the tunnel interface an IP address that is the same as that of interface Loopback0. **Note** This command is not effective until Lookback0 has been configured with an IP address.
Step 8	**destination** *ip-address* **Example:** `switch(config-if-te)# destination 10.3.3.3`	Specifies the IP address of the device where the tunnel will terminate. This address should be the router ID of the device that is the NHOP or NNHOP of LSPs to be protected.
Step 9	**path-option** [**protect**] *preference-number* {**dynamic** | **explicit** {**identifier** *id* | **name** *name*} [**verbatim**]} [**lockdown**] [**bandwidth** *kbps*] [**attributes** *listname*] **Example:** `switch(config-if-te)# path-option 10 explicit avoid-protected-link`	Configures a path option for an MPLS TE tunnel.

Assigning Backup Tunnels to a Protected Interface

You can assign one or more backup tunnels to a protected interface by entering the commands on the node that will be the headend of the backup tunnel (the node whose downstream link or node might fail). The node on which you enter these commands must be a supported platform.

Note This procedure is required for configuring fast reroute link and node protection.

Note You must configure the interface to have an IP address and enable the MPLS TE tunnel feature.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **mpls traffic-eng backup-path tunnel-te** *interface*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface** *type slot/port* **Example:** `switch(config)# interface Ethernet 5/0` `switch(config-if)#`	Moves configuration to the physical interface level, directing subsequent configuration commands to the specific physical interface identified by the *type* value. The *slot* and *port* identify the slot and port being configured. The interface must be a supported interface.
Step 3	**mpls traffic-eng backup-path tunnel-te** *interface* **Example:** `switch(config-if)# mpls traffic-eng backup-path tunnel-te 2`	Allows LSPs going out this interface to use this backup tunnel if there is a link or node failure. **Note** You can enter this command multiple times to associate multiple backup tunnels with the same protected interface.

Associating Backup Bandwidth and Pool Type with a Backup Tunnel

You can associate the backup bandwidth with a backup tunnel and designate the type of LSP that can use a backup tunnel.

Note You can use this optional procedure to configure fast reroute link and node protection.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel-te** *number*
3. **backup-bw** *bandwidth*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface tunnel-te** *number* **Example:** `switch(config)# interface tunnel-te 2` `switch(config-if-te)#`	Enters interface configuration mode for the specified tunnel.
Step 3	**backup-bw** *bandwidth* **Example:** `switch(config-if-te)# backup-bw sub-pool 1000`	Associates bandwidth with a backup tunnel.

Configuring Backup Bandwidth Protection

You can configure backup bandwidth protection.

Note You can use this optional procedure to configure fast reroute link and node protection.

Prerequisites

You must have the MPLS TE feature enabled (see the "Configuring MPLS TE" section on page 10-139).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **mpls traffic-eng configuration**
3. **fast-reroute** [**bw-protect**]
4. **fast-reroute backup-prot-preemption** [**optimize-bw**]

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls traffic-eng configuration** **Example:** `switch(config)# mpls traffic-eng configuration` `switch(config-te)`	Enters MPLS TE configuration mode.
Step 3	**fast-reroute** [**bw-protect**] **Example:** `switch(config-te)# fast-reroute bw-protect`	Enables an MPLS TE tunnel to use an established backup tunnel in the event of a link or node failure. The **bw-protect** keyword gives an LSP priority for using backup tunnels with bandwidth protection.
Step 4	**fast-reroute backup-prot-preempt** [**optimize-bw**] **Example:** `switch(config-te)# fast-reroute backup-prot-preempt optimize-bw`	Changes the backup protection preemption algorithm from minimizing the number of LSPs that are demoted to minimizing the amount of bandwidth that is wasted.

Verifying the MPLS TE Fast Reroute Link and Node Protection Configuration

To determine if FRR has been configured correctly, enter the **show mpls traffic-eng tunnels brief** command and the **show ip rsvp sender detail** command. If you created LSPs and performed the required configuration tasks but do not have operational backup tunnels (that is, the backup tunnels are not up or the LSPs are not associated with those backup tunnels), enter the **show mpls traffic-eng tunnel fast-reroute** command.

Command	Purpose
show mpls traffic-eng tunnels brief	Displays backup tunnel status.
show ip rsvp sender detail	Displays LSPs protection from the appropriate backup tunnels.
show mpls traffic-eng tunnel fast-reroute	Displays whether MPLS TE FRR node protection has been enabled and that a certain type of LSP can use a backup tunnel.
show mpls traffic-eng tunnels backup	Displays whether fast reroute backup is configured. Enter the command on the router where the backup tunnels originate.

Command	Purpose
show mpls traffic-eng tunnel fast-reroute	Displays the LSPs that are protected.
show ip rsvp reservation	When entered at the headend of the primary LSP, this command displays the status of FRR (local protection) at each hop this LSP traverses. The per-hop information is collected in the Record Route Object (RRO) that travels with the Resv message from the tail to the head.

The following example shows how to verify that the backup tunnels are up:

```
switch# show mpls traffic-eng tunnels brief

Signalling Summary:
    LSP Tunnels Process:            running
    Passive LSP Listener:           running
    RSVP Process:                   running
    Forwarding:                     enabled
    Periodic reoptimization:        every 3600 seconds, next in 2822 seconds
    Periodic FRR Promotion:         Not Running
    Periodic auto-bw collection:    every 300 seconds, next in 206 seconds
TUNNEL NAME                     DESTINATION      UP IF     DOWN IF   STATE/PROT
nxti-rt-1_t1                    10.0.0.32        -         Eth2/2    up/up
nxti-rt-1_t5                    0.0.0.0          -         unknown   admin-down
nxti-rt-1_t123                  192.168.20.123   -         unknown   admin-down
nxti-rt-1_t12345                192.168.20.1     -         unknown   admin-down
nxti-rt-1_t12345_12346          192.168.20.1     -         unknown   admin-down
nxti-rt-1_t12345_12346          192.168.20.1     -         unknown   admin-down
Displayed 6 (of 5) heads, 0 (of 0) midpoints, 0 (of 0) tails
```

The following example shows how to verify that the LSPs are protected by the appropriate backup tunnels. The output was when the command was entered at the PLR before a failure.

```
switch# show ip rsvp sender detail

Tun Dest:   10.0.0.32  Tun ID: 1  Ext Tun ID: 10.0.0.30
Tun Sender: 10.0.0.30  LSP ID: 850
Last Refresh Send: Nrefresh [18 seconds ago, status ok]
sent:     on Ethernet2/2
Session Attr:
    Setup Prio: 7, Holding Prio: 7
    Flags: (0x4)
    SE style    Session Name:nxti-rt-1_t1
ERO: (incoming)
    60.0.0.2 (Strict IPv4 Prefix, 8 bytes, prefix /32)
    61.0.0.1 (Strict IPv4 Prefix, 8 bytes, prefix /32)
    61.0.0.2 (Strict IPv4 Prefix, 8 bytes, prefix /32)
    10.0.0.32 (Strict IPv4 Prefix, 8 bytes, prefix /32)
ERO: (outgoing)
    60.0.0.2 (Strict IPv4 Prefix, 8 bytes, prefix /32)
    61.0.0.1 (Strict IPv4 Prefix, 8 bytes, prefix /32)
    61.0.0.2 (Strict IPv4 Prefix, 8 bytes, prefix /32)
    10.0.0.32 (Strict IPv4 Prefix, 8 bytes, prefix /32)
RRO:    Empty
Class-Type: None
 TSPEC: T=2, L=36: Version=0, 7 words
  Token bucket frag (service_id=1, 6 words)
     param id=127, flags=0, 5 words
     avg rate=12500 bytes/sec, depth=1000 bytes
     peak rate=12500 bytes/sec
     min unit=40 bytes, max unit=500 bytes
Fast-reroute:
```

```
  Outbound FRR: No backup tunnel selected
  Inbound FRR: not active
  Path ID handle: 0x98f00352.
  Incoming policy: Accepted.   Policy source(s): Default
 Status:  Proxied
  Output on Ethernet2/2.
Policy status: Forwarding. Policy Handle: 0x98f00353
    Outstanding report.
```

The following example shows how to verify that MPLS TE FRR node protection has been enabled and that a certain type of LSP can use a backup tunnel.

Note Enter the **clear ip rsvp hello instance counters** command to verify the following:
- MPLS TE FRR node protection has been enabled.
- A certain type of LSP can use a backup tunnel.

```
switch# show mpls traffic-eng tunnel fast-reroute
```

If LDP is not enabled, separate prefix items are not shown because all prefixes then use a single rewrite. To confirm that a particular IP prefix is FRR protected, even though it is not shown in this display, enter it within the **show mpls forwarding-table** *ip-address* **detail** command. The final line of the display will tell whether that prefix is protected:

```
switch# show mpls forwarding-table 10.0.0.11 detail

Local    Outgoing    Prefix         Bytes tag    Outgoing     Next Hop
tag      tag or VC   or Tunnel Id   switched     interface
Tun hd   Untagged    10.0.0.11/32   48           Eth5/0       point2point
          MAC/Encaps=4/8, MTU=1520, Tag Stack{22}
          48D18847 00016000
          No output feature configured
          Fast Reroute Protection via (Tu0, outgoing label 12304)
```

The following example shows how to verify that the fast reroute backup is configured.

Enter the command on the router where the backup tunnels originate.

```
switch# show mpls traffic-eng tunnels backup

Router_t578
  LSP Head, Tunnel578, Admin: up, Oper: up
  Src 10.55.55.55, Dest 10.88.88.88, Instance 1
  Fast Reroute Backup Provided:
    Protected i/fs: PO1/0, PO1/1, PO3/3
    Protected lsps: 1
    Backup BW: any pool unlimited; inuse: 100 kbps
Router_t5710
  LSP Head, Tunnel5710, Admin: admin-down, Oper: down
  Src 10.55.55.55, Dest 10.7.7.7, Instance 0
  Fast Reroute Backup Provided:
    Protected i/fs: PO1/1
    Protected lsps: 0
    Backup BW: any pool unlimited; inuse: 0 kbps
Router_t5711
  LSP Head, Tunnel5711, Admin up, Oper: up
  Src 10.55.55.55,, Dest 10.7.7.7, Instance 1
  Fast Reroute Backup Provided:
    Protected i/fs: PO1/0
    Protected lsps: 2
    Backup BW: any pool unlimited; inuse: 6010 kbps
```

The following example shows how to display the LSPs that are protected.

Note Enter the **clear ip rsvp hello instance counters** command to verify the following:
- MPLS TE FRR node protection has been enabled.
- A certain type of LSP can use a backup tunnel.

The output was entered at the headend of a primary LSP. Entering the command at the headend of the primary LSP shows the status of FRR (local protection) at each hop that this LSP traverses. The per-hop information is collected in the Record Route Object (RRO) that travels with the Resv message from the tail to the head.

Note the following about the primary LSP:

- It has protection that uses a NHOP backup tunnel at its first hop.
- It has protection and is actively using an NHOP backup tunnel at its second hop.
- It has no local protection at its third hop.

The RRO display shows the following information for each hop:

- Whether local protection is available (that is, whether the LSP has selected a backup tunnel).
- Whether local protection is in use (that is, whether the LSP is currently using its selected backup tunnel).
- Whether the selected backup tunnel is an NHOP or NNHOP backup tunnel.
- Whether the backup tunnel used at this hop provides bandwidth protection.

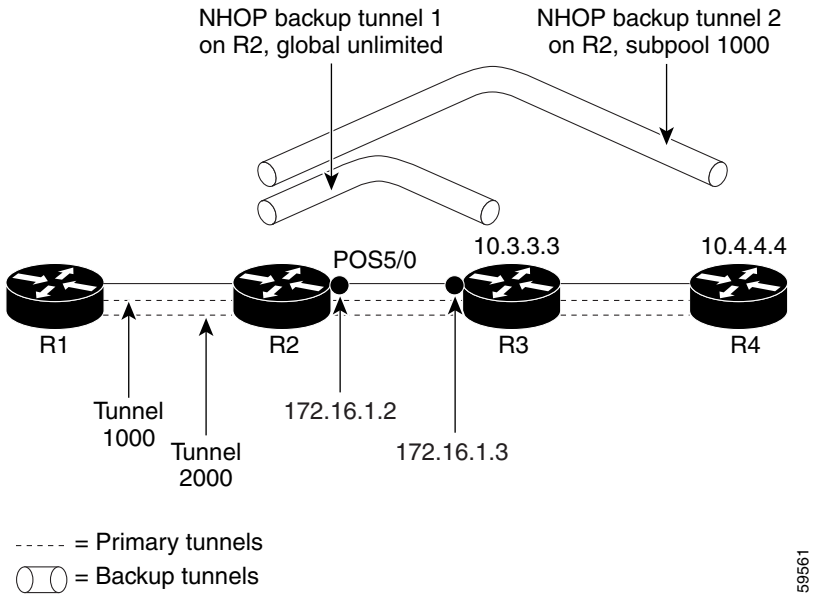
```
switch# show ip rsvp reservation detail

Reservation:
  Tun Dest: 10.1.1.1  Tun ID: 1  Ext Tun ID: 172.16.1.1
  Tun Sender: 172.16.1.1  LSP ID: 104
  Next Hop: 172.17.1.2 on Eth1/0
  Label: 18 (outgoing)
  Reservation Style is Shared-Explicit, QoS Service is Controlled-Load
  Average Bitrate is 0 bits/sec, Maximum Burst is 1K bytes
  Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  RRO:
    172.18.1.1/32, Flags:0x1 (Local Prot Avail/to NHOP)
      Label subobject: Flags 0x1, C-Type 1, Label 18
    172.19.1.1/32, Flags:0x0 (Local Prot Avail/In Use/Has BW/to NHOP)
      Label subobject: Flags 0x1, C-Type 1, Label 16
    172.19.1.2/32, Flags:0x0 (No Local Protection)
      Label subobject: Flags 0x1, C-Type 1, Label 0
  Resv ID handle: CD000404.
  Policy: Accepted. Policy source(s): MPLS/TE
```

Configuration Examples of MPLS TE Fast Reroute Link and Node Protection

This section includes the following topics:

The examples relate to Figure 18-8.

Figure 18-8 Backup Tunnels



Example: Enabling Fast Reroute for all Tunnels

The following example shows how to enable fast reroute for all tunnels.

On router R1, enter interface configuration mode for each tunnel to be protected (Tunnel 1000 and Tunnel 2000). Enable these tunnels to use a backup tunnel in case of a link or node failure along their paths.

Tunnel 1000 uses 10 units of bandwidth from the subpool.

Tunnel 2000 uses 5 units of bandwidth from the global pool. The bandwidth protection desired bit has been set by specifying **bw-prot** in the **fast-reroute** command.

```
switch(config)# interface tunnel-te 1000
switch(config-if-te)# fast-reroute
switch(config-if-te)# bandwidth sub-pool 10

switch(config)# interface Tunnel2000
switch(config-if-te)# fast-reroute bw-prot
switch(config-if-te)# bandwidth 5
```

Example: Creating an NHOP Backup Tunnel

The following example shows how to create an NHOP backup tunnel.

On router R2, create an NHOP backup tunnel to R3. This backup tunnel should avoid using the link 172.1.1.2.

```
switch(config)# mpls traffic-eng configuration
switch(config-te)# explicit-path name avoid-protected-link
switch(config-te-expl-path)# exclude-address 172.1.1.2
Explicit Path name avoid-protected-link:
____1: exclude-address 172.1.1.2
switch(cfg-ip_expl-path)# exit
switch(config-te)# exit
switch(config)# interface tunnel-te 1
switch(config-if-te)# ip unnumbered loopback0
switch(config-if-te)# destination 10.3.3.3
switch(config-if-te)# path-option 10 explicit avoid-protected-link
```

Example: Creating an NNHOP Backup Tunnel

The following example shows how to create an NNHOP backup tunnel.

On router R2, create an NNHOP backup tunnel to R4. This backup tunnel should avoid R3.

```
switch(config)# mpls traffic-eng configuration
switch(config-te)# explicit-path name avoid-protected-node
switch(config-te-expl-path)# exclude-address 10.3.3.3
Explicit Path name avoid-protected-node:
____1: exclude-address 10.3.3.3
switch(cfg-ip_expl-path)# exit
switch(config-te)# exit

switch(config)# interface tunnel-te 2
switch(config-if-te)# ip unnumbered loopback0
switch(config-if-te)# destination 10.4.4.4
switch(config-if-te)# path-option 10 explicit avoid-protected-node
```

Example: Assigning Backup Tunnels to a Protected Interface

The following example shows how to assign backup tunnels to a protected interface.

On router R2, associate both backup tunnels with interface Ethernet 5/0:

```
switch(config)# interface Ethernet 5/0
switch(config-if)# mpls traffic-eng backup-path tunnel 1
switch(config-if)# mpls traffic-eng backup-path tunnel 2
```

Example: Associating the Backup Bandwidth and Pool Type with Backup Tunnels

The following example shows how to associate the backup bandwidth and pool with backup tunnels.

Backup tunnel 1 is to be used only by LSPs that take their bandwidth from the global pool. It does not provide bandwidth protection. Backup tunnel 2 is to be used only by LSPs that take their bandwidth from the subpool. Backup tunnel 2 provides bandwidth protection for up to 1000 units.

```
switch(config)# interface tunnel-te 1
switch(config-if-te)# backup-bw global-pool Unlimited

switch(config)# interface tunnel-te 2
switch(config-if-te)# backup-bw sub-pool 1000
```

Example: Configuring Backup Bandwidth Protection

The following example shows how to configure backup bandwidth protection.

Note This global configuration is required only to change the backup protection preemption algorithm from minimizing the number of LSPs that are demoted to minimizing the amount of bandwidth that is wasted.

```
switch(config-if-te)# fast-reroute bw-protect
switch(config-te)# fast-reroute backup-prot-preemption optimize-bw
```

Example: Configuring RSVP Hello

Hello must be configured both globally on the router and on the specific interface on which you need FRR protection. To configure Hello, use the following configuration commands:

- **ip rsvp signalling hello** (configuration)—Enables Hello globally on the router.
- **ip rsvp signalling hello** (interface)—Enables Hello on an interface where you need FRR protection.

The following configuration commands are optional:

- **ip rsvp signalling hello dscp**—Sets the differentiated services code point (DSCP) value that is in the IP header of the Hello message.
- **ip rsvp signalling hello refresh misses**—Specifies how many acknowledgments that a node can miss in a row before the node considers that communication with its neighbor is down.
- **ip rsvp signalling hello refresh interval**—Configures the Hello request interval.
- **ip rsvp signalling hello statistics**—Enables Hello statistics on the router.

For configuration examples, see the Hello command descriptions in the “Command Reference” section of *MPLS Traffic Engineering (TE): Link and Node Protection, with RSVP Hellos Support*.

Troubleshooting Tips

This section provides the following troubleshooting information:

- LSPs Do Not Become Active; They Remain Ready, page 18-299

LSPs Do Not Become Active; They Remain Ready

At a PLR, LSPs transition from Ready to Active if one of the following events occurs:

- Primary interface goes down—If the primary interface (LSP's outbound interface) goes down and the LSP is ready to use a backup tunnel, the LSP transitions to the active state causing its data to flow over the backup tunnel. On some platforms and interface types, there is a fast interface-down logic that detects this event very quickly. On other platforms where this logic does not exist, detection time is slower. On such platforms, you might enable BFD.
- Hellos detect next hop is down—If Hellos are enabled on the primary interface (LSP's outbound interface), and the LSP's next hop is no longer reachable, the next hop is declared down. This event will cause the LSP to begin actively using its backup tunnel. Notice that a next hop is declared down even if the primary interface does not go down. For example, if the next hop stops responding due to a reboot or a software or hardware problem, Hellos trigger the LSPs using this next hop to switch to their backup tunnels. Hellos can also help trigger FRR on interfaces such as Gigabit Ethernet where the interface remains up but is unusable (due to a lack of link-layer liveness detection mechanisms).

Primary Tunnel Does Not Select Backup Tunnel That Is Up

If a backup tunnel is up, but it is not selected as a backup tunnel by the primary tunnel (LSP), enter the following commands for the backup tunnel:

- **shutdown**
- **no shutdown**

Note If you change the status of a backup tunnel, the backup tunnel selection algorithm is rerun for the backup tunnel. LSPs that have currently selected (ready to use) that backup tunnels are disassociated from it, and then reassociated with that backup tunnel or another backup tunnel, which is generally harmless and usually results in mapping the same LSPs to that backup tunnel. However, if any LSPs are actively using that backup tunnel, shutting down the backup tunnel tears down those LSPs.

Enhanced RSVP Commands Display Useful Information

The following RSVP commands have been enhanced to display information that can be helpful when you are examining the FRR state or troubleshooting FRR:

- **show ip rsvp request**—Displays the upstream reservation state (information that is related to the Resv messages that this node will send upstream).
- **show ip rsvp reservation**—Displays information about Resv messages received.
- **show ip rsvp sender**—Displays information about path messages being received.

These commands show the control plane state; they do not show the data state. They show information about RSVP messages (Path and Resv) used to signal LSPs. For information about the data packets being forwarded along LSPs, use the **show mpls forwarding** command.

RSVP Hello Detects When a Neighboring Node Is Not Reachable

The RSVP Hello feature enables RSVP nodes to detect when a neighboring node is not reachable. Use this feature when notification of link-layer failures is not available and unnumbered links are not used or when the failure detection mechanisms provided by the link layer are not sufficient for timely node failure detection. You must configure Hello both globally on the router and on the specific interface to be operational.

Hello Instances Have Not Been Created

If Hello instances have not been created, do the following:

- Determine if RSVP Hello has been enabled globally on the router. Enter the **ip rsvp signalling hello** (configuration) command.
- Determine if RSVP Hello has been enabled on an interface that the LSPs traverse. Enter the **ip rsvp signalling hello** (interface) command.

Verify that at least one LSP has a backup tunnel by displaying the output of the **show ip rsvp sender** command. A value of “Ready” indicates that a backup tunnel has been selected.

“No entry at index” (error may self-correct, RRO may not yet have propagated from downstream node of interest)” Error Message is Displayed at the Point of Local Repair

FRR relies on a RRO in Resv messages arriving from downstream. Routers that receive path messages with the SESSION_ATTRIBUTE bit indicating that the LSP is fast-reroutable should include an RRO in the corresponding Resv messages.

If an LSP is configured for FRR, but the Resv arriving from a downstream router contains an incomplete RRO, the “No entry at index (error may self-correct, RRO may not yet have propagated from downstream node of interest)” message is displayed. An incomplete RRO is when the NHOP or the NNHOP did not include an entry in the RRO.

This error typically means that backup tunnels to the NHOP or the NNHOP cannot be selected for this LSP because there is insufficient information about the NHOP or NNHOP due to a missing RRO entry.

There are valid circumstances in which this situation occurs temporarily and the problem self corrects. If subsequent Resv messages arrive with a complete RRO, you should ignore the error message.

To determine whether the error has been corrected, display the RRO in Resv messages by entering the **clear ip rsvp hello instance counters** command. Use an output filter keyword to display only the LSP of interest.

"Couldn't get rsbs" (error may self-correct when Resv arrives)" Error Message is Displayed at the Point of Local Repair (PLR)

The PLR cannot select a backup tunnel for an LSP until a Resv message has arrived from downstream. When this error occurs, it typically means that something is wrong. For example, no reservation exists for this LSP.

There are valid circumstances in which this error message is displayed and there is no need for concern. One such circumstance is when an LSP experiences a change before any Resv message has arrived from downstream. Changes can cause a PLR to try to select a backup tunnel for an LSP, and the selection will fail (causing this error message) if no Resv message has arrived for this LSP.

Additional References for MPLS TE Fast Reroute Link and Node Protection

The following sections provide references related to the fast reroute link and node protection feature.

Related Documents

Related Topic	Document Title
MPLS TE commands	*Cisco NX-OS Multiprotocol Label Switching Command Reference*
RSVP	*Cisco NX-OS Quality of Service Commands*
IS-IS	*Configuring a Basic IS-IS Network*
OSPF	*Cisco NX-OS Unicast Routing Protocols Command Reference* *Configuring OSPF*
Link protection	*MPLS TE: Link and Node Protection, with RSVP Hellos Support (with Fast Tunnel Interface Down Detection)*
Configuration of MPLS TE tunnels	*MPLS Traffic Engineering: Interarea Tunnels*

MIBs

MIBs	MIBs Link
• CISCO-IETF-FRR-MIB • MPLS TE-STD-MIB	To locate and download MIBs for selected platforms, Cisco NX-OS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for MPLS TE Fast Reroute Link and Node Protection

Table 18-3 lists the release history for this feature.

Table 18-3 Feature History for MPLS TE Fast Reroute Link and Node Protection

Feature Name	Releases	Feature Information
Fast reroute link and node protection	5.2(1)	This feature was introduced.

Configuring MPLS Quality of Service

This chapter describes how to configure Multiprotocol Label Switching (MPLS) quality of service (QoS) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS QoS

MPLS QoS enables you to provide differentiated types of service across an MPLS network. Differentiated types of service satisfy a range of requirements by supplying the service specified for each packet.

Note QoS allows you to classify the network traffic, police and prioritize the traffic flow, and provide congestion avoidance. For general information about QoS in Cisco NX-OS, see the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*.

This section includes the following topics:

MPLS QoS Terminology

This section defines some MPLS QoS terminology:

- *Class of Service* (CoS) refers to three bits in an 802.1Q header that are used to indicate the priority of the Ethernet frame as it passes through a switched network. The CoS bits in the 802.1Q header are commonly referred to as the 802.1p bits. To maintain QoS when a packet traverses both Layer 2 and Layer 3 domains, the type of service (ToS) and CoS values can be mapped to each other.
- *Classification* is the process used for selecting traffic to be marked for QoS.
- *Differentiated Services Code Point* (DSCP) is the first six bits of the ToS byte in the IP header. DSCP is only present in an IP packet.
- Disposition is the process of removing or popping one or more MPLS labels by the edge label switch router (LSR) when the packet is leaving an MPLS domain.
- *E-LSP* is a label switched path (LSP) on which nodes infer the QoS treatment for MPLS packets exclusively from the experimental (EXP) bits in the MPLS header. Because the QoS treatment is inferred from the EXP (both class and drop precedence), several classes of traffic can be multiplexed onto a single LSP (use the same label). A single LSP can support up to eight classes of traffic because the EXP field is a 3-bit field.
- *EXP bits* define the QoS treatment (per-hop behavior) that a node should give to a packet. It is the equivalent of the DiffServ Code Point (DSCP) in the IP network. A DSCP defines a class and drop precedence. The EXP bits are generally used to carry all the information encoded in the IP DSCP. In some cases, however, the EXP bits are used exclusively to encode the dropping precedence.
- *Frames* carry traffic at Layer 2. Layer 2 frames carry Layer 3 packets.
- *IP precedence* is the three most significant bits of the ToS byte in the IP header.

- Imposition is the process of adding or imposing one or more MPLS labels by the edge LSR when the packet is entering an MPLS domain.
- *QoS tags* are prioritization values carried in Layer 3 packets and Layer 2 frames. A Layer 2 CoS label can have a value ranging between zero for low priority and seven for high priority. A Layer 3 IP precedence label can have a value ranging between zero for low priority and seven for high priority. IP precedence values are defined by the three most significant bits of the 1-byte ToS byte. A Layer 3 DSCP label can have a value between 0 and 63. DSCP values are defined by the six most significant bits of the 1-byte IP ToS field.
- *LERs* (label edge routers) are devices that impose and dispose of labels upon packets; LERs are also referred to as Provider Edge (PE) routers.
- *LSRs* (label switching routers) are devices that forward traffic based upon labels present in a packet; LSRs are also referred to as Provider (P) routers.
- *Marking* is the process of setting a Layer 3 DSCP value in a packet. Marking is also the process of choosing different values for the MPLS EXP field to mark packets so that they have the priority that they require during periods of congestion.
- *MQC* is the Cisco Modular QoS command line interface (MQC) framework, which is a modular and highly extensible framework for deploying QoS.
- *Packets* carry traffic at Layer 3.
- *Policing* is limiting bandwidth used by a flow of traffic. Policing can mark or drop traffic.
- Swapping is the process of replacing one or more MPLS labels by the LSR within an MPLS domain.

MPLS QoS Features

These topics describe the following MPLS QoS features that are supported in an MPLS network:

MPLS Experimental Field

Setting the MPLS experimental (EXP) field value satisfies the requirement of operators who do not want the value of the IP precedence field modified within IP packets transported through their networks.

By choosing different values for the MPLS EXP field, you can mark packets so that packets have the priority that they require during periods of congestion.

By default, the three most significant bits of the DSCP are copied into the MPLS EXP field during imposition.You can mark the MPLS EXP bits with an MPLS QoS policy.

Trust

For received Layer 2 or Layer 3 MPLS packets, the router usually trusts the EXP value in the received topmost label. None of the following have any effect on MPLS packets:

- Interface trust state
- Port CoS value

Classification

Classification is the process that selects the traffic to be marked. Classification partitions traffic into multiple priority levels or classes of service. Traffic classification is the primary component of class-based QoS provisioning. The router makes classification decisions based on the EXP bits in the received topmost label of the received MPLS packets (after a policy is installed). For more information, see the "Configuring a Class Map to Classify MPLS Packets" section on page 19-319.

Policing and Marking

Policing causes traffic that exceeds the configured rate to be discarded or marked down to a higher drop precedence. Marking is a way to identify packet flows to differentiate them. Packet marking allows you to partition your network into multiple priority levels or classes of service.

The MPLS QoS policing and marking features that you can implement depend on the received traffic type and the forwarding operation applied to the traffic. See the "Configuring a Policy Map" section on page 19-319 for information.

Preserving IP ToS

The router automatically preserves the IP type of service (ToS) during all MPLS operations including imposition, swapping, and disposition.You do not need to enter a command to save the IP ToS.

EXP Mutation

You can configure a named egress EXP mutation map to mutate the internal DSCP-derived EXP value before it is used as the egress EXP value. You can attach egress EXP mutation maps to these interface types:

- LAN port subinterfaces
- Layer 3 VLAN interfaces
- Layer 3 LAN ports

You cannot attach egress EXP mutation maps to these interface types:

- Layer 2 LAN ports (ports that are configured with the **switchport** command)

MPLS DiffServ Tunneling

The router uses MPLS DiffServ tunneling to provide QoS transparency from one edge of a network to the other edge of the network. See the "MPLS DiffServ Tunneling Modes" section on page 19-314 for more information.

MQC CLI

All policing and marking features available for MPLS QoS are managed from the modular QoS command-line interface (CLI). The modular QoS CLI (MQC) allows you to define traffic classes (class maps), create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

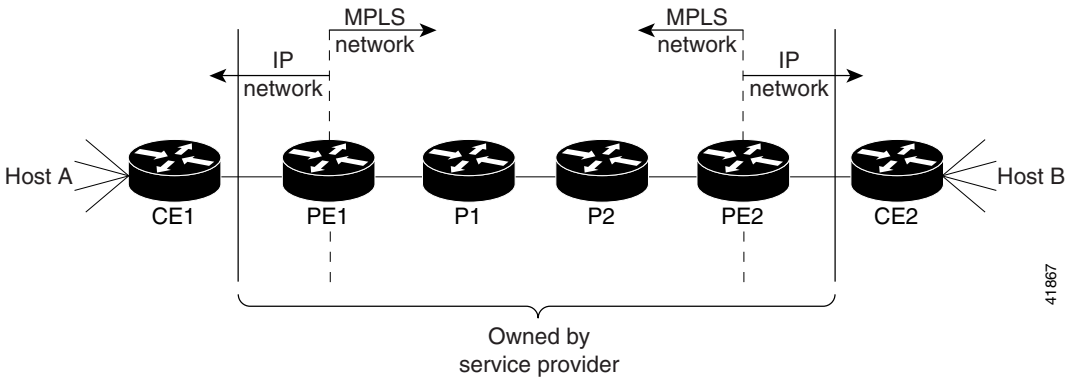
Note For detailed information on configuring QoS in NX-OS using the MQC, see the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*.

Topology and Roles

This section describes the roles of network devices in implementing MPLS QoS.

Figure 19-1 shows an MPLS network of a service provider that connects two sites of a customer network.

Figure 19-1 MPLS Network Connecting Two Sites of a Customer's IP Network



The network is bidirectional, but for the purpose of this document, the packets move left to right.

In Figure 19-1, the symbols have the following meanings:

- CE1—Customer equipment 1
- PE1—Service provider ingress LER
- P1—LSR within the core of the network of the service provider
- P2—LSR within the core of the network of the service provider
- PE2—Service provider egress LER
- CE2—Customer equipment 2

Note PE1 and PE2 are at the boundaries between the MPLS network and the IP network.

This section includes the following topics:

- LERs at the Ingress Edge of an MPLS Network, page 19-309
- LSRs at the Core of an MPLS Network, page 19-309

- LERs at the Egress Edge of an MPLS Network, page 19-310

LERs at the Ingress Edge of an MPLS Network

Note Incoming labels are aggregate or nonaggregate. The aggregate label indicates that the arriving MPLS or MPLS VPN packet must be switched through an IP lookup to find the next hop and the outgoing interface. The nonaggregate label indicates that the packet contains the IP next-hop information.

At the ingress edge of an MPLS network, LERs process packets as follows:

1. Layer 2 or Layer 3 traffic enters the edge of the MPLS network at the edge LER (PE1).
2. The router receives the traffic from the input interface and uses the 802.1p bits or the IP ToS bits to perform any classification, marking, and policing to derive the new EXP bits. For classification of incoming IP packets, the input service policy can also use access control lists (ACLs).
3. For each incoming packet, the router performs a lookup on the IP address to determine the next-hop router.
4. The appropriate label is pushed (imposition) into the packet, and the EXP value resulting from the QoS decision is copied into the MPLS EXP field in the label header.
5. The router forwards the labeled packets to the appropriate output interface for processing.
6. The router also forwards the 802.1p bits or the IP ToS bits to the output interface. The 802.1p bits or the IP ToS bits can be remarked by EXP bits in the topmost label.
7. The labeled packets (marked by EXP) are sent to the core MPLS network.

LSRs at the Core of an MPLS Network

At the core of an MPLS network, LSRs process packets as follows:

1. Incoming MPLS-labeled packets (and 802.1p bits or IP ToS bits) from an edge LER (or other core device) arrive at the core LSR.
2. The router receives the traffic from the input interface and uses the EXP bits to perform classification, marking, and policing.
3. The router performs a table lookup to determine the next-hop LSR.
4. An appropriate label is placed (swapped) into the packet and the MPLS EXP bits are copied into the label header.
5. The router forwards the labeled packets to the appropriate output interface for processing.
6. The router also forwards the 802.1p bits or the IP ToS bits to the output interface.
7. The outbound packet is differentiated by the MPLS EXP field for marking or policing.
8. The labeled packets (marked with EXP) are sent to another LSR in the core MPLS network or to an LER at the output edge.

Note Within the network, there is no IP precedence field for the queueing algorithm to use because the packets are MPLS packets. The packets remain MPLS packets until they arrive at PE2, the provider edge router.

LERs at the Egress Edge of an MPLS Network

At the egress edge of an MPLS network, LERs process packets as follows:

1. MPLS-labeled packets (and 802.1p bits or IP ToS bits) from a core LSR arrive at the egress LER (PE2) from the MPLS network backbone.
2. The router pops the MPLS labels (disposition) from the packets. Aggregate and nonaggregate labels are classified with the EXP value by default.
3. For aggregate labels, the router performs a lookup on the IP address to determine the packet's destination; the router then forwards the packet to the appropriate output interface for processing. For nonaggregate labels, forwarding is based on the label. By default, nonaggregate labels are popped at the penultimate-hop router (next to last), not the egress PE router.
4. The router also forwards the 802.1p bits or the IP ToS bits to the output interface.
5. The packets are differentiated according to the 802.1p bits or the IP ToS bits and treated accordingly.

Note The MPLS EXP bits allow you to specify the QoS for an MPLS packet. The IP precedence and DSCP bits allow you to specify the QoS for an IP packet.

MPLS QoS Classification at the Edges and the Core

This section includes the following topics:

IP to MPLS

This section provides information about QoS features for LERs at the ingress (CE-to-PE) and egress (PE-to-CE) edges for MPLS and MPLS VPN networks. Both MPLS and MPLS VPN support general MPLS QoS features.

The router provides the following MPLS QoS capabilities at the IP-to-MPLS edge:

- Assigning an EXP value based on the **policy-map** command
- Marking an EXP value using a policy
- Policing traffic using a policy

This section includes the following topics:

Classification for IP to MPLS

The router ingress and egress policies for IP traffic classify traffic on the original received IP using **match** commands for IP precedence, IP DSCP, and IP ACLs.

After the router applies the QoS policies, it assigns the internal DSCP. The router then assigns the EXP value based on the internal DSCP-to-EXP global map for the labels that it imposes. If more than one label is imposed, the EXP value is the same in each label. The router preserves the original IP ToS when the MPLS labels are imposed.

The router assigns the egress CoS based on the internal DSCP-to-CoS global map. If the default internal DSCP-to-EXP and the internal DSCP-to-CoS maps are consistent, then the egress CoS has the same value as the imposed EXP.

If the ingress port receives both IP-to-IP and IP-to-MPLS traffic, you use classification to separate the two types of traffic. For example, if the IP-to-IP and IP-to-MPLS traffic have different destination address ranges, you can classify traffic on the destination address, apply IP ToS policies to the IP-to-IP traffic, and apply a policy (that marks or sets the EXP value in the imposed MPLS header) to the IP-to-MPLS traffic. See the following two examples:

- A router policy to mark IP ToS sets the internal DSCP—If it is applied to all traffic, then for IP-to-IP traffic, the egress port rewrites the CoS (derived from the internal DSCP) to the IP ToS byte in the egress packet. For IP-to-MPLS traffic, the router maps the internal DSCP to the imposed EXP value.
- A router policy to mark MPLS EXP sets the internal DSCP—If it is applied to all traffic, then for IP-to-IP traffic, the egress port rewrites the IP ToS according to the ingress IP policy (or trust). The CoS is mapped from the ToS. For IP-to-MPLS traffic, the router maps the internal DSCP to the imposed EXP value.

Classification for IP-to-MPLS Mode MPLS QoS

MPLS QoS at the ingress to PE1 supports the following:

- Matching on IP precedence or DSCP values or filtering with an access group
- The **set mpls experimental imposition** and **police** commands

MPLS QoS at the egress of PE1 supports the **mpls experimental topmost** command.

Classification at IP-to-MPLS Ingress Port

Classification for IP to MPLS is the same as for IP to IP. Port classification is based on the received Layer 2 802.1Q CoS value.

Classification at IP-to-MPLS Egress Port

Port classification is based on the received EXP value, and the egress CoS value is mapped from that value.

If the egress port is a trunk, the ports copy the egress CoS into the egress 802.1Q field.

MPLS to MPLS

This section provides information about MPLS QoS features for LSRs at the core (MPLS to MPLS) for MPLS and MPLS VPN networks.

MPLS QoS at the MPLS core supports the following:

- Per-EXP policing based on a service policy
- Copying the input topmost EXP value into the newly imposed EXP value
- Optional EXP mutation (changing of EXP values on an interface edge between two neighboring MPLS domains) on the egress boundary between MPLS domains

- Optional propagation of the topmost EXP value into the underlying EXP value when popping the topmost label from a multilabel stack.

This section includes the following topics:

Classification for MPLS to MPLS

For received MPLS packets, the router trusts the EXP value in the topmost label.

Note The MPLS QoS ingress and egress policies for MPLS traffic classify traffic on the EXP value in the received topmost label when you enter the **match mpls experimental** command.

MPLS QoS maps the EXP value to the internal DSCP using the EXP-to-DSCP global map. What the router does next depends on whether it is swapping labels, imposing a new label, or popping a label:

- Swapping labels—When swapping labels, the router preserves the EXP value in the received topmost label and copies it to the EXP value in the outgoing topmost label. The router assigns the egress CoS using the internal DSCP-to-CoS global map. If the DSCP global maps are consistent, then the egress CoS is based on the EXP in the outgoing topmost label.

  The router can mark down out-of-profile traffic using the **police** command's **exceed** and **violate** actions. It does not mark in-profile traffic, so the **conform** action must be transmitted and the **set** command cannot be used. If the router is performing a markdown, it uses the internal DSCP as an index into the internal DSCP markdown map. The router maps the result of the internal DSCP markdown to an EXP value using the internal DSCP-to-EXP global map. The router rewrites the new EXP value to the topmost outgoing label and does not copy the new EXP value to the other labels in the stack. The router assigns the egress CoS using the internal DSCP-to-CoS global map. If the DSCP maps are consistent, then the egress CoS is based on the EXP value in the topmost outgoing label.

- Imposing an additional label—When imposing a new label onto an existing label stack, the router maps the internal DSCP to the EXP value in the imposed label using the internal DSCP-to-EXP map. It then copies the EXP value in the imposed label to the underlying swapped label. The router assigns the egress CoS using the internal DSCP-to-CoS global map. If the DSCP maps are consistent, the egress CoS is based on the EXP value in the imposed label.

  The router can mark in-profile and mark down out-of-profile traffic. After it marks the internal DSCP, the router uses the internal DSCP-to-EXP global map to map the internal DSCP to the EXP value in the newly imposed label. The router then copies the EXP in the imposed label to the underlying swapped label. The router assigns the egress CoS using the internal DSCP-to-CoS global map. Therefore, the egress CoS is based on the EXP in the imposed label.

- Popping a label—When popping a label from a multi-label stack, the router preserves the EXP value in the exposed label. The router assigns the egress CoS using the internal DSCP-to-CoS global map. If the DSCP maps are consistent, then the egress CoS is based on the EXP value in the popped label.

- If EXP propagation is configured for the egress interface, the router maps the internal DSCP to the EXP value in the exposed label using the DSCP-to-EXP global map. The router assigns the egress CoS using the internal DSCP-to-CoS global map. If the DSCP maps are consistent, the egress CoS is based on the EXP value in the exposed label.

Classification for MPLS-to-MPLS QoS

MPLS QoS at the ingress to P1 or P2 supports the following:

- Matching with the **mpls experimental topmost** command
- The **set mpls experimental imposition**, **police**, and **police** with **set imposition** commands

MPLS QoS at the egress of P1 or P2 supports matching with the **mpls experimental topmost** command.

Classification at MPLS-to-MPLS Ingress Port

Port classification is based on the ingress CoS from the router.

Classification at MPLS-to-MPLS Egress Port

Port classification is based on the egress CoS value from the router.

If the egress port is a trunk, the LAN ports copy the egress CoS into the egress 802.1Q field.

MPLS to IP

This section provides information about QoS features for LERs at the egress (PE-to-CE) edges for MPLS and MPLS VPN networks. Both MPLS and MPLS VPN support general MPLS QoS features.

MPLS QoS supports these capabilities at the MPLS-to-IP edge:

- Option to propagate the EXP value into IP DSCP on exit from an MPLS domain per egress interface
- Option to use IP service policy on the MPLS-to-IP egress interface

This section includes the following topics:

Classification for MPLS to IP

The router assigns the internal DSCP (internal priority that the router assigns to each frame) based on the QoS result. The QoS result is affected by the following:

- Default trust EXP value
- Number of VPNs
- Explicit NULL use
- QoS policy

The router does one of the following:

- Preserves the underlying IP ToS
- Rewrites the IP ToS by a value derived from the EXP-to-DSCP global map
- Changes the IP ToS to any value derived from the egress IP policy

In all cases, egress queueing is based on the final IP ToS from the DSCP-to-CoS map.

For incoming MPLS packets on the PE-to-CE ingress, the router supports MPLS classification and ingress IP policies. The PE-to-CE traffic from the MPLS core is classified or policed on egress as IP.

Classification for MPLS-to-IP MPLS QoS

MPLS QoS at the ingress to PE2 supports matching on the EXP value and the **police** command.

MPLS QoS at the egress of PE2 supports matching on the IP precedence, DSCP values, or filtering with an access group and the **police** command.

Classification at MPLS-to-IP Ingress Port

Ingress port classification is based on the EXP value.

Classification at MPLS-to-IP Egress Port

Classification for MPLS to IP is the same as it is for IP to IP.

Egress port classification is based on the egress CoS.

If the egress port is a trunk, the ports copy the egress CoS into the egress 802.1Q field.

Note For MPLS to IP, egress IP ACL or QoS is not effective on the egress interface if the egress interface has MPLS IP (or tag IP) enabled. The exception is a VPN CAM hit, where the packet is classified on egress as IP.

MPLS DiffServ Tunneling Modes

Tunneling provides QoS the ability to be transparent from one edge of a network to the other edge of the network. A tunnel starts where there is label imposition. A tunnel ends where there is label disposition; that is, where the label is removed from the stack, and the packet goes out as an MPLS packet with a different per-hop behavior (PHB) layer underneath or as an IP packet with the IP PHB layer.

For the MPLS router, there are two ways to forward packets through a network:

- Short Pipe or Pipe mode—EXP marking implemented in the core does not propagate to the packet ToS byte. In Short Pipe mode, the egress PE router uses the original Layer 3 packet marking instead of the marking used by the intermediate provider (P) routers. In Pipe mode, the egress PE router uses the Layer 2 marking of the intermediate provider (P) routers.
- Uniform mode—EXP marking implemented in the core is propagated to the underlying ToS byte. In Uniform mode, you can manipulate the marking in the IP packet to reflect the operator's QoS marking in the core. This mode provides consistent QoS classification and marking throughout the network including the CE and core routers.

Both tunneling modes affect the behavior of edge and penultimate LSRs where labels are put onto packets and removed from packets. They do not affect label swapping at intermediate routers. An operator can choose different types of tunneling modes for each customer.

For additional information, see *MPLS DiffServ Tunneling Modes* at this URL:
http://www.cisco.com/en/US/docs/ios/12_2t/12_2t13/feature/guide/ftdtmode.html

MPLS QoS and HA

The Cisco NX-OS architecture and High Availability (HA) infrastructure provide support for feature components to be restarted and resume operations transparently to other services on the device and on neighboring devices. This feature allows for continuous operation and zero data loss during planned software changes and unplanned software failures.

MPLS QoS supports these Cisco NX-OS HA features:

- Nonstop forwarding (NSF)
- Stateful HA

MPLS QoS supports these Cisco NX-OS HA technologies to allow NSF and stateful HA:

- Stateful process restart
- Stateful switch over (SSO)
- In-Service Software Upgrade (ISSU)

Licensing Requirements for MPLS QoS

Product	License Requirement
Cisco NX-OS	MPLS QoS requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS QoS

MPLS QoS has the following prerequisites:

- Your network must support Multiprotocol Label Switching (MPLS)
- Your network must support at least one of the following Interior Gateway (IGP) protocols:
  - Intermediate System-to-Intermediate System (IS-IS)
  - Open Shortest Path First (OSPF)

Guidelines and Limitations for MPLS QoS

MPLS QoS has the following configuration guidelines and limitations:

- There is no global command to enable or disable MPLS QoS. If no QoS polices are configured, the switch follows the default behavior described in this section.
- For IP-to-MPLS imposition when the received packet is an IP packet, the following applies:
  - When no QoS polices are configured, the EXP value is based on the received IP ToS.
- For MPLS-to-MPLS operations, the following applies:
  - When swapping and no QoS polices are configured, the EXP value is based on the original EXP value (in the absence of EXP mutation).

  - When swapping and QoS is queuing only, the EXP value is based on the original EXP value (in the absence of EXP mutation).
  - When imposing an additional label and no QoS polices are configured, the EXP value is based on the original EXP value (in the absence of EXP mutation).
  - When imposing an additional label and QoS is queuing only, the EXP value is based on the original EXP value (in the absence of EXP mutation).
  - When popping one label and no QoS polices are configured, the EXP value is based on the underlying EXP value.
  - When popping one label and QoS is queuing only, the EXP value is based on the underlying EXP value.
- For classifying MPLS packets with class maps, the following applies:
  - The **match mpls experimental** command specifies the name of an EXP field value to be used as the match criterion against which packets are checked to determine if they belong to the class specified by the class map.
  - To use the **match mpls experimental** command, you must first enter the **class-map** command to specify the name of the class whose match criteria you want to establish. After you identify the class, you can use the **match mpls experimental** command to configure its match criteria.
  - If you specify more than one command in a class map, only the last command entered applies. The last command overrides the previously entered commands.
- When configuring a policy map, the following applies:
  - You can attach only one policy map of a given type to an interface in a specific direction.
- When setting the EXP value on all imposed labels, the following applies:
  - Use the **set mpls experimental imposition** command during label imposition. This command sets the MPLS EXP field on all imposed label entries.
  - The **set mpls experimental imposition** command is supported only on input interfaces (imposition).
  - The **set mpls experimental imposition** command does not mark the EXP value directly; instead, it marks the internal DSCP that is mapped to EXP through the internal DSCP-to-EXP global map.
  - It is important to note that classification (based on the original received IP header) and marking (done to the internal DSCP) do not distinguish between IP-to-IP traffic and IP-to-MPLS traffic. The commands that you use to mark IP ToS and mark EXP have the same result as when you mark the internal DSCP.
  - Use the **set mpls experimental imposition** command to set the pushed label entry value to a value that is different from the default value during label imposition.
  - You optionally can use the **set mpls experimental imposition** command with the IP precedence, DSCP field, or QoS IP ACL to set the value of the MPLS EXP field on all imposed label entries.
  - When imposing labels onto the received IP traffic, you can mark the EXP field using the **set mpls experimental imposition** command.
- When using the **police** command to configure a policy map, the following applies:
  - With MPLS, the **exceed-action** *action* command and the **violate-action** *action* command work similarly to IP usage. The packet might get dropped or the EXP value is marked down.

  - When swapping labels for received MPLS traffic, you can mark down out-of-profile traffic using the **police** command **exceed-action policed-dscp-transmit** and **violate-action policed-dscp-transmit** keywords. The router does not mark in-profile traffic; when marking down out-of-profile traffic, the router marks the outgoing topmost label. The router does not propagate the marking down through the label stack.
  - You can use the **police** command to set the pushed label entry value to a value that is different from the default value during label imposition.
  - When imposing labels onto the received IP traffic, you can mark the EXP field using the **conform-action set-mpls-exp-imposition-transmit** keywords.
- Before you downgrade from Cisco NX-OS Release 5.2 or later to an earlier Cisco NX-OS release, clear the QoS MIB and MPLS QoS defaults using the **clear qos mpls-snmp** command. Otherwise, the downgrade might fail.
- When you configure a partial queueing policy such as the **set mpls-exp-topmost cos table exp-cos-map** command on a default queue, only the MPLS configuration is programmed in the hardware. Other parameters in the hardware, such as queue limits, bandwidth, and thresholds, retain the previously applied configuration. When the line card is reloaded, previously applied configuration parameters are not retained because new policy parameters are absent for the queue limits, bandwidth, and thresholds. As a result, some hardware parameters might be different before and after the reload. To address this problem, define parameters like queue limits, bandwidth, and thresholds for class maps along with the MPLS configuration to copy the EXP bits to cos.

Default Settings for MPLS QoS

Table 19-1 lists the default settings for MPLS QoS parameters.

Table 19-1 Default Settings for MPLS QoS

Parameters	Default
Port CoS value	0 (zero)
Port-based or VLAN-based QoS	Port-based
EXP to DSCP map (DSCP set from EXP values)	EXP 0 = DSCP 0 EXP 1 = DSCP 8 EXP 2 = DSCP 16 EXP 3 = DSCP 24 EXP 4 = DSCP 32 EXP 5 = DSCP 40 EXP 6 = DSCP 48 EXP 7 = DSCP 56
IP precedence to DSCP map (DSCP set from IP precedence values)	IP precedence 0 = DSCP 0 IP precedence 1 = DSCP 8 IP precedence 2 = DSCP 16 IP precedence 3 = DSCP 24 IP precedence 4 = DSCP 32 IP precedence 5 = DSCP 40 IP precedence 6 = DSCP 48 IP precedence 7 = DSCP 56

Table 19-1 Default Settings for MPLS QoS (continued)

Parameters	Default
DSCP to EXP map (EXP set from DSCP values)	DSCP 0–7 = EXP 0 DSCP 8–15 = EXP 1 DSCP 16–23 = EXP 2 DSCP 24–31 = EXP 3 DSCP 32–39 = EXP 4 DSCP 40–47 = EXP 5 DSCP 48–55 = EXP 6 DSCP 56–63 = EXP 7
Marked-down DSCP from DSCP map	Marked-down DSCP value equals original DSCP value (no mark down)
EXP mutation map	No mutation map by default
Policers	None
Policy maps	None
MPLS flow mask in NetFlow table	Label and EXP value
MPLS core QoS	There are four possibilities at the MPLS core QoS: • Swapping—The incoming EXP field is copied to the outgoing EXP field. • Swapping and imposition—The incoming EXP field is copied to both the swapped EXP field and the imposed EXP field. **Note** If there is a service policy with a set for EXP field, its EXP field is placed into the imposed label and also into the swapped label. • Disposition of topmost label—The exposed EXP field is preserved. • Disposition of only label—The exposed IP DSCP is preserved.
MPLS to IP edge QoS	Preserve the exposed IP DSCP

Configuring MPLS QoS

Note You can configure MPLS QoS commands inside a config session.

This section include the following topics:

- Configuring a Class Map to Classify MPLS Packets, page 19-319
- Configuring a Policy Map, page 19-319
- Creating a Table Map, page 19-322

Configuring a Class Map to Classify MPLS Packets

You can configure a class map.

SUMMARY STEPS

1. **configure terminal**
2. **[no] class-map [type qos] [match-any]** *class-map-name*
3. **[no] match [not] mpls experimental topmost** *exp-list*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**[no] class-map [type qos] [match-any]** *class-map-name* **Example:** switch(config)# class-map type qos match-any Class2 switch(config-cmap-qos)#	Defines a class map, enters class-map configuration mode, and specifies the packet matching criteria as follows: • **match-any**—(Optional) Specifies that if a packet matches any of the policies of this class map, the class map is applied to the packet.
Step 3	**[no] match [not] mpls experimental topmost** *exp-list* **Example:** switch(config-cmap-qos)# match mpls experimental topmost 2, 5-7	Specifies that the packets should be matched (or not) on the 3-bit experimental (EXP) field in the outermost (topmost) MPLS label in the MPLS header as follows: • ***exp-list***—The list can contain values and ranges. Values can range from 0 to 7.

Configuring a Policy Map

This section includes the following topics:

Configuring a Policy Map to Set the EXP Value on All Imposed Labels

You can configure a policy map to set the EXP value on all imposed labels.

To set the value of the MPLS EXP field on all imposed label entries, use the **set mpls experimental imposition** command in QoS policy-map class configuration mode. To disable the setting, use the **no** form of this command.

SUMMARY STEPS

1. **configure terminal**

2. [**no**] **policy-map [type qos]** *policy-map-name*
3. [**no**] **class** [**type qos**] {*class-map-name* | **class-default**} [**insert-before** *class-map-name*]
4. **set mpls experimental imposition** *mpls-exp-value*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	[**no**] **policy-map** [**type qos**] *policy-map-name* **Example:** switch(config)# policy-map Policy1 switch(config-pmap-qos)#	Defines a class map and enters class-map configuration mode.
Step 3	[**no**] **class** [**type qos**] {*class-map-name* | **class-default**} **[insert-before *class-map-name*]** **Example:** switch(config-pmap-qos)# class Class2 switch(config-pmap-c-qos)	Specifies the class name of the policy to create or change: • ***class-map-name***—The name of the class to configure or modify the policy. • **class-default**—A predefined class to which traffic is directed if that traffic does not match any of the match criteria in the configured class maps.
Step 4	**set mpls experimental imposition** *mpls-exp-value* **Example:** switch(config-pmap-c-qos)# set mpls experimental imposition 3	Sets the value of the MPLS experimental (EXP) field on all imposed label entries.

Configuring a Policy Map Using the Police Command

Policing is a function in the router hardware that provides the ability to rate limit a particular traffic class to a specific rate. The router supports aggregate policing. Microflow policing is not supported.

Aggregate policing meters all traffic that ingresses into a port, regardless of the different source, destination, protocol, source port, or destination port.

Note For additional information about using the **police** command, see the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **policy-map [type qos]** *policy-map-name*
3. [**no**] **class** [**type qos**] {*class-map-name* | **class-default**}
4. [**no**] **police** [**cir**] {*x* [**bps** | **kbps** | **mbps** | **gbps**] | **percent** *x-percent*} [[**bc**] *bc* [**bytes** | **kbytes** | **mbytes** | **ms** | **us**]] [**pir** *y* [**bps** | **kbps** | **mbps** | **gbps**] | **percent** *y-percent*] [[**be**] *be* [**bytes** | **kbytes** | **mbytes** | **ms** | **us**]] [**conform** *conform-action* [**exceed** *exceed-action* [**violate** *violate-action*]]]

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	[**no**] **policy-map** [**type qos**] *policy-map-name* **Example:** switch(config)# policy-map Policy1 switch(config-pmap-qos)	Defines a class map and enters class-map configuration mode.
Step 3	[**no**] **class** [**type qos**] {*class-map-name* | **class-default**} **Example:** switch(config-pmap-qos)# class Class2 switch(config-pmap-c-qos)	Specifies the class name of the policy to create or change: • ***class-map-name***—The name of the class to configure or modify the policy. • **class-default**—A predefined class to which traffic is directed if that traffic does not match any of the match criteria in the configured class maps.
Step 4	[**no**] **police** [**cir**] {*x* [**bps** | **kbps** | **mbps** | **gbps**] | **percent** *x-percent*} *[[**bc**] bc* [**bytes** | **kbytes** | **mbytes** | **ms** | **us**]] [**pir** *y* [**bps** | **kbps** | **mbps** | **gbps**] | **percent** *y-percent] [[**be**] be* [**bytes** | **kbytes** | **mbytes** | **ms** | **us**]] **[conform** *conform-action* [**exceed** *exceed-action* [**violate** *violate-action*]]] **Example:** switch(config-pmap-c-qos)# police cir 256000 conform set-mpls-exp-topmost-transmit violate drop	Specifies that a class of traffic should have a maximum rate imposed on it, and if that rate is exceeded, an immediate action must be taken. In addition to the conform actions described in the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*, MPLS QoS provides two conform actions: • **set-mpls-exp-imposition-transmit**—Sets the EXP field on all imposed labels to a specified value and transmits the packet. • **set-mpls-exp-topmost-transmit**—Sets the EXP field on the outer (topmost) label to a specified value and transmits the packet.

Configuring a Policy Map Using Table Maps

You can use the system-defined table maps to perform marking in the **set** and **police** policy map class commands.

Note For general information about using table maps, see the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x.*

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **policy-map [type qos]** *policy-map-name*
3. [**no**] **class** [**type qos**] **{***class-map-name* **| class-default}**
4. [**no**] **set** *to-field from-field* **table** *table-map-name*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	[**no**] **policy-map** [**type qos**] *policy-map-name* **Example:** switch(config)# policy-map Policy1 switch(config-pmap-qos)	Defines a class map and enters class-map configuration mode.
Step 3	[**no**] **class** [**type qos**] {*class-map-name* | **class-default**} **Example:** switch(config-pmap-qos)# class Class2 switch(config-pmap-c-qos)	Specifies the class name of the policy to create or change: • *class-map-name*—The name of the class to configure or modify the policy. • **class-default**—A predefined class to which traffic is directed if that traffic does not match any of the match criteria in the configured class maps.
Step 4	**set** *to-field from-field* **table** *table-map-name* **Example:** switch(config-pmap-c-qos)# set cos mpls-exp-topmost table dscp-cos-map	Defines a mapping of values between two QoS fields. In addition to the *to-field* and *from-field* arguments described in the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*, MPLS QoS provides the following fields: • *to-field:* – **mpls-exp-topmost**—The EXP field on the outer (topmost) label. • *from-field:* – **mpls-exp-imposition**—The EXP field to be applied on all imposed labels. – **mpls-exp-topmost**—The EXP field to be applied on the outer (topmost) label. **Note** The *table-map-name* argument can be the name of a system-defined table map or an existing table map defined using the **table-map** command.

Creating a Table Map

You can define a table that remaps QoS values for use in the **set** and **police** policy map class commands.

Note For additional information about table maps, see the *Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*.

SUMMARY STEPS

1. **configure terminal**

2. **table-map** *table-map-name*
3. **from** *number* **to** *number*
4. Repeat step 3 to complete table.

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**table-map** *table-map-name* **Example:** switch(config)# table-map TableMap1 switch(config-tmap)	Creates a table map and enters table map configuration mode.
Step 3	**from** *number* **to** *number* **Example:** switch(config-tmap)# from 1 to 1 switch(config-tmap)# from 2 to 1 switch(config-tmap)# from 3 to 2 switch(config-tmap)# from 4 to 2 ...	Maps one number to another. The range of *number* is from 0 to 63. This step can repeat up to 64 times.
Step 4	Repeat Step 3 to complete table.	—

Verifying the MPLS QoS Configuration

To display the MPLS QoS configuration, perform the following task:

Command	Purpose
show policy-map type qos interface *interface*	Displays the statistics that show the packets matched for every class on that interface in the given direction.

Need supported show commands, with examples.

Configuration Examples for MPLS QoS

This section includes the following configuration examples:

- Example: Configuring a Class Map to Classify MPLS Packets, page 19-324
- Example: Configuring a Policy Map to Set the EXP Value on All Imposed Labels, page 19-324
- Example: Configuring a Policy Map Using the Police Command, page 19-324
- Example: Configuring a Policy Map Using Table Maps, page 19-324

Example: Configuring a Class Map to Classify MPLS Packets

The following example shows how to create a class map to match EXP field values 2 and 5 through 7 on the outer (topmost) label of an MPLS header:

```
switch# configure terminal
switch(config)# class-map Class2
switch(config-cmap-qos)# match mpls experimental topmost 2, 5-7
```

Example: Configuring a Policy Map to Set the EXP Value on All Imposed Labels

The following example shows how to create a policy map to set the EXP field to a value of 3 on all imposed labels of an MPLS header:

```
switch# configure terminal
switch(config)# policy-map Policy1
switch(config-pmap-qos)# class Class2
switch(config-pmap-c-qos)# set mpls experimental imposition 3
```

Example: Configuring a Policy Map Using the Police Command

The following example shows how to create a policy map to enforce a committed data rate of 256000 bps. If the data rate is in conformance, the router sets the EXP field of the outer (topmost) label of the MPLS header. If the data rate is exceeded, the router drops packets.

```
switch# configure terminal
switch(config)# policy-map Policy1
switch(config-pmap-qos)# class Class2
switch(config-pmap-c-qos)# police cir 256000 conform set-mpls-exp-topmost-transmit violate
drop
```

Example: Configuring a Policy Map Using Table Maps

The following example shows how to create a policy map that maps CoS values of 0 to 3 to an EXP field value of 1 and CoS values of 4 to 7 to an EXP field value of 6. The EXP value is then written to the outer (topmost) label of the MPLS header.

```
switch# configure terminal
switch(config)# table-map TableMap1
switch(config-tmap)# from 0 to 1
switch(config-tmap)# from 1 to 1
switch(config-tmap)# from 2 to 1
switch(config-tmap)# from 3 to 1
switch(config-tmap)# from 4 to 6
switch(config-tmap)# from 5 to 6
switch(config-tmap)# from 6 to 6
switch(config-tmap)# from 7 to 6
switch(config-tmap)# exit
switch(config)# policy-map Policy1
switch(config-pmap-qos)# class Class2
switch(config-pmap-c-qos)# set cos mpls-exp-topmost table TableMap1
```

Additional References for MPLS QoS

The following sections provide references related to the MPLS QoS feature.

Related Document

Related Topic	Document Title
Cisco NX-OS QoS configuration	*Cisco Nexus 7000 Series NX-OS Quality of Service Configuration Guide, Release 6.x*
Cisco NX-OS MPLS QoS commands	*Cisco Nexus 7000 Series NX-OS Quality of Service Command Reference*
Cisco NX-OS MPLS commands	*Cisco NX-OS Multiprotocol Label Switching Command Reference*

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for MPLS QoS

Table 19-2 lists the release history for this feature.

Table 19-2 Feature History for MPLS QoS

Feature Name	Releases	Feature Information
MPLS QoS	5.2(1)	This feature was introduced.

Configuring MPLS Layer 3 VPNs

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Layer 3 virtual private networks (VPNs) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS Layer 3 VPNs

An MPLS Layer 3 VPN consists of a set of sites that are interconnected by an MPLS provider core network. At each customer site, one or more customer edge (CE) routers or Layer 2 switches attach to one or more provider edge (PE) routers.

This section includes the following topics:

MPLS Layer 3 VPN Definition

MPLS-based Layer 3 VPNs are based on a peer model that enables the provider and the customer to exchange Layer 3 routing information. The provider relays the data between the customer sites without direct customer involvement.

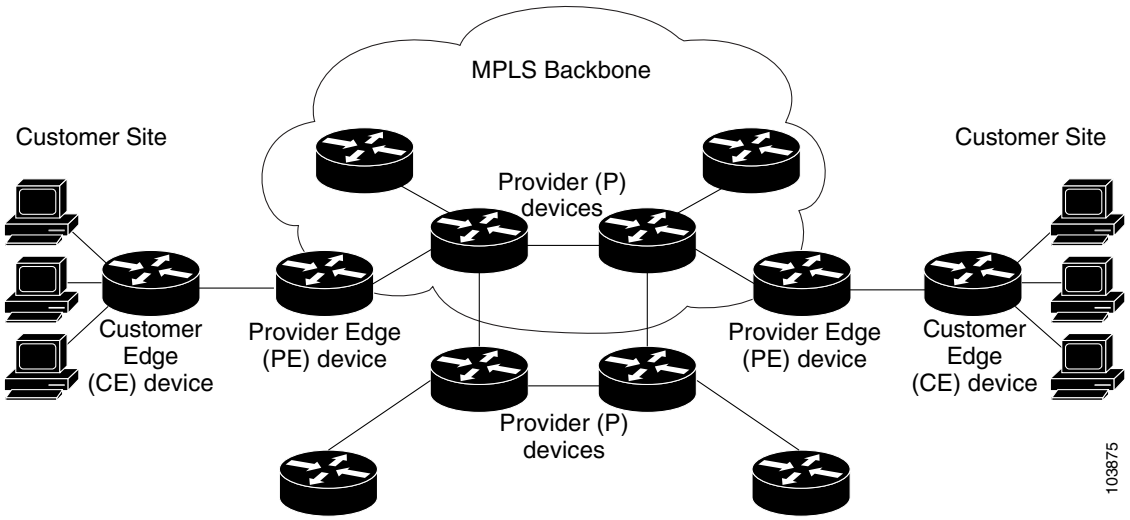
When you add a new site to an MPLS Layer 3 VPN, you must update the provider edge router that provides services to the customer site.

MPLS Layer 3 VPNs include the following components:

- Provider (P) router—A router in the core of the provider network. P routers run MPLS switching and do not attach VPN labels (an MPLS label in each route assigned by the PE router) to routed packets. P routers forward packets based on the Label Distribution Protocol (LDP).
- Resource Reservation Protocol (RSVP) traffic engineering (TE)— A protocol that assigns a label to the egress PE router.
- Provider edge (PE) router—A router that attaches the VPN label to incoming packets that are based on the interface or subinterface on which they are received. A PE router attaches directly to a CE router.
- Customer edge (CE) router—An edge router on the network of the provider that connects to the PE router on the network. A CE router must interface with a PE router.

Figure 20-1 shows a basic MPLS Layer 3 VPN.

Figure 20-1 Basic MPLS Layer 3 VPN Terminology



How an MPLS Layer 3 VPN Works

MPLS Layer 3 VPN functionality is enabled at the edge of an MPLS network. The PE router performs the following tasks:

- Exchanges routing updates with the CE router
- Translates the CE routing information into VPN routes
- Exchanges Layer 3 VPN routes with other PE routers through the Multiprotocol Border Gateway Protocol (MP-BGP)

How VRF Tables Work in an MPLS Layer 3 VPN

Each Layer 3 VPN is associated with one or more virtual routing and forwarding (VRF) instance. A VRF defines the VPN membership of a customer site that is attached to a PE router. A VRF consists of the following components:

- An IP routing table
- A set of interfaces that use the forwarding table
- A set of rules and routing protocol parameters that control the information that is included in the routing table

A one-to-one relationship does not necessarily exist between customer sites and VPNs. A site can be a member of multiple VPNs. Typically, a CE router at a site can associate with only one VRF. The VRF of the CE router contains all the routes that are available to the site from the VPNs of which the VRF is a member.

Packet forwarding information is stored in the IP routing table for each VRF. A separate set of routing tables is maintained for each VRF. These tables prevent information from being forwarded outside a VPN and also prevent packets that are outside a VPN from being forwarded to a router within the VPN.

VPN Route Distribution and Route Targets

The distribution of VPN routing information is controlled through VPN route targets that are implemented by BGP extended communities. VPN routing information is distributed as follows:

- When a VPN route that is learned from a CE router is injected into BGP, a list of VPN route target extended community attributes is associated with the VPN route. Typically, the list of route target community extended values is set from an export list of route targets associated with the VRF from which the route was learned.
- An import list of route target extended communities is associated with each VRF. The import list defines route target extended community attributes that a route must have in order for the route to be imported into the VRF. For example, if the import list for a particular VRF includes route target extended communities A, B, and C, then any VPN route that carries any of those route target extended communities—A, B, *or* C—is imported into the VRF.

Route Leaking and Importing Routes from the Default VRF

You can import IP prefixes from the global routing table (the default VRF) into any other VRF by using an import policy. The VRF import policy uses a route map to specify the prefixes to be imported into a VRF. The policy can import IPv4 and IPv6 unicast prefixes.

Note Routes in the BGP default VRF can be imported directly. Any other routes in the global routing table should be redistributed into BGP first.

IP prefixes are defined as match criteria for the import route map through standard route policy filtering mechanisms. For example, you can create an IP prefix list or an as-path filter to define an IP prefix or IP prefix range and use that prefix list or as-path filter in a match clause for the route map. Prefixes that pass through the route map are imported into the specified VRF using the import policy. IP prefixes that are imported into a VRF through this import policy cannot be reimported into another VPN VRF.

The maximum number of prefixes that can be imported from the default VRF is controlled by a limit that you configure.

VRF Route Table Limits

You can configure a limit to the number of routes that are accepted and installed into a VRF routing table to prevent overloading the PE router. This limit applies only to dynamic routing protocols and not to static or connected routes. Alternately, when you use eBGP as the PE-CE protocol, you can configure a per-neighbor maximum prefix limit.

VPN ID and Route Distinguisher

You use an MPLS VPN ID to identify a VPN but not to control route distribution or routing updates. You assign the same VPN ID to all routers in the provider network that service the VPN. The VPN ID format is specified in RFC 2685.

The route distinguisher (RD) is an eight-byte value that is combined with the IPv4 or IPv6 prefix learned by the PE router to create a globally unique address.

6VPE

The IPv6 PE router over MPLS Virtual Private Network (6VPE) feature is an extension of Layer 3 VPNs that support VPN connectivity for IPv6 sites over an MPLS/IPv4 provider core network. The VPN-IPv6 address is formed by adding an 8-byte RD to a 16-byte IPv6 address, which results in a 24-byte VPN-IPv6 address. 6VPE uses VRF tables to assign the forwarding information at the PE and uses the IPv6 address family. BGP supports the VPN-IPv6 address family. This address family supports both per-prefix and per-VRF label allocation modes.

6VPE prepends the IPv4 next-hop address with ::FFFF: to create the IPv4-mapped IPv6 address for the next hop that is advertised.

Note MPLS Layer 3 load balancing is supported for 6VPE but is not supported with per-VRF label allocation.

VPN IPv4 sites often use private addressing for their addressing plan. These addresses do not need to be registered, and they are not routable on the public network. Whenever a host within a private site needs to access a public domain, it goes through a device that finds a public address on its behalf, such as a network address translator or an application proxy.

Due to the larger address space available with IPv6, the easiest approach to IPv6 addressing is to use IPv6 global addresses for the private addressing plan. Another approach is to use unique local addresses (ULAs). ULAs are easy to filter at site boundaries based on their local scope. ULAs are Internet service provider (ISP)-independent and can be used for communications inside a site without any permanent or intermittent Internet connectivity.

In 6VPE, ULAs are treated as regular global addresses. The router configuration filters ULA prefixes to prevent them from appearing in the public domain. Link-local addresses on the peer are not announced by BGP (IPv6 or IPv6 VPN) speakers.

A host within a private site that needs to access a public domain can do so through an IPv6 application proxy (such as a web proxy for accessing web pages), which accesses the public resource, on behalf of the host, with a global routable address, or the host can use a public address of its own. In the latter case, if you have deployed ULAs, the IPv6 host also is configured with a routable global address. A source address selection algorithm is used to select one or the other, based on the destination address.

BGP PIC

BGP Prefix Independent Convergence (PIC) achieves subsecond convergence in the forwarding plane for BGP IP and Layer 3 VPN routes in various cases of BGP next-hop network reachability failures. BGP PIC has two categories: PIC Core and PIC Edge. PIC Core ensures fast convergence for BGP routes when there is a link or node failure in the core that causes a change in the IGP reachability to a remote BGP next-hop address. PIC Edge ensures fast convergence to a precomputed BGP backup path when an external (eBGP) edge link or an external neighbor node fails.

IPv4, VPNv4, 6PE, and VPNv6 (6VPE) support PIC Core with the following constraints:

- For both IP and MPLS core, convergence for Internet routes is prefix-independent on the order of BGP next hops.
- With per-VRF label allocation, VPN route convergence is also prefix-independent on the order of BGP next hops. That is, when a path to a remote PE changes, convergence is determined by the number of VRFs on that PE.
- With per-prefix label allocation, route convergence is not prefix-independent. Convergence moves to the order of VPN routes that are advertised by a remote PE if a failure or change occurs in the reachability to that PE.

Note PIC edge is not supported.

BGP Distribution of VPN Routing Information

A PE router can learn an IP prefix from the following sources:

- A CE router by static configuration
- A directly connected network
- A BGP session with the CE router
- A routing protocol exchange with the CE router

After the PE router learns the IP prefix, the PE can conditionally export the prefix into a VPN prefix by combining it with an 8-byte route distinguisher. The generated prefix is a member of the VPN-IPv4 or the VPN-IPv6 address family. It uniquely identifies the customer address, even if the customer site is using globally nonunique (unregistered private) IP addresses. You configure the route distinguisher that generates the VPN-IPv4 or VPN-IPv6 prefix on the VRF on the PE router.

BGP distributes reachability information for VPN prefixes for each VPN. BGP communication takes place at two levels:

- Within an autonomous system using interior BGP (iBGP)
- Between autonomous systems using external BGP (eBGP)

PE-PE or PE-RR (route reflector) sessions are iBGP sessions, and PE-CE sessions are eBGP sessions. BGP propagates reachability information for VPN-IPv4 and VPN-IPv6 prefixes among PE routers by using BGP multiprotocol extensions (see RFC 2283, *Multiprotocol Extensions for BGP-4*). The BGP multiprotocol extensions define support for address families other than IPv4. When you use the extensions, you ensure that the routes for a given VPN are learned only by other members of that VPN. This process enables members of the VPN to communicate with each other.

In an Enhanced Interior Gateway Routing Protocol (EIGRP) PE-CE environment, when an EIGRP internal route is redistributed into BGP by one PE, then back into EIGRP by another PE, the originating router ID for the route is set to the router ID of the second PE. This process replaces the original internal router ID.

Note The BGP minimum route advertisement interval (MRAI) value for all iBGP and eBGP sessions is zero and is not configurable.

BGP Next-Hop Address Tracking

See the "Configuring Advanced BGP" chapter of the *Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide* for information.

MPLS Forwarding

Based on routing information in the VRF IP routing table, the router forwards packets to their destination using MPLS.

A PE router binds a label to each customer prefix that is learned from a CE router and includes the label in the network reachability information for the prefix that it advertises to other PE routers. When a PE router forwards a packet that it received from a CE router across the provider network, it labels the packet with the label learned from the destination PE router. When the destination PE router receives the labeled packet, it removes the label and uses it to direct the packet to the correct CE router. Label forwarding across the provider backbone is based on either dynamic label switching or traffic engineered paths. A customer data packet carries two levels of labels when it traverses the backbone:

- The top label directs the packet to the correct PE router.
- The second label indicates how that PE router should forward the packet to the CE router.

Site of Origin

The site of origin prevents routing loops when you have a multihomed VPN site. Routes learned from the same site are tagged with the same site-of-origin value that is configured at the PE on all the PE-CE links to the same site. Routes with a particular site-of-origin value are never readvertised back to a CE with the same site-of-origin value configured at the PE-CE link. This process prevents a CE router from relearning routes that originated from the same site. BGP and EIGRP use site of origin to prevent loops.

You can override the autonomous system number (ASN) of a site with the ASN of the provider. This feature is often used with the site of origin to identify the site where a route originated and prevent routing loops between routers within a VPN.

Site of Origin and EIGRP

When EIGRP is used as the PE-CE routing protocol, EIGRP uses BGP extended communities to carry the EIGRP vector metric, AS number, and other information to recreate the EIGRP internal routes with the original attributes across the VPN cloud. EIGRP external routes or routes from a different autonomous system are recreated as external routes.

EIGRP uses site of origin to prevent routing loops when you have a multihomed VPN site. You must configure the site of origin for EIGRP-based PE routes that are learned from the CE. We recommend you use the site of origin for CE routers for better performance.

You might want to disable the BGP best path cost community option in a multihomed VPN site and use the internal routes to fully utilize all PE-CE links. The default behavior is that only one PE-CE link is used and the other PE-CE links serve as backup links.

OSPF Sham Link

Although Open Shortest Path First (OSPF) PE-CE connections assume that the only path between two client sites is across the MPLS Layer 3 VPN backbone, backdoor paths between VPN sites might exist. If these sites belong to the same OSPF area, the router always chooses the path over a backdoor link because OSPF prefers intra-area paths to interarea paths. (PE routers advertise OSPF routes that they learned over the VPN backbone as interarea paths.)

To reestablish the desired path selection over the MPLS Layer 3 VPN backbone, you must create an additional OSPF intra-area (logical) link between ingress and egress VRFs on the relevant PE routers. This link is called a sham link. A sham link is required between any two VPN sites that belong to the same OSPF area and share an OSPF backdoor link. If no backdoor link exists between the sites, no sham link is required. When a sham link is configured between PE routers, the PEs can populate the VRF routing table with the OSPF routes learned over the sham link. Because OSPF sees the sham link as an intra-area link between PE routers, an OSPF creates an adjacency and triggers a database exchange (for the particular OSPF process) across the link. The PE router can then flood LSAs between sites from across the MPLS VPN backbone and create intra-area connectivity.

OSPF LSA Throttling

OSPF LSA throttling is enabled by default and allows faster OSPF convergence (in milliseconds). You can control the generation (sending) of LSAs, control the receiving interval, and provide a dynamic mechanism to slow down the frequency of LSA updates in OSPF during times of network instability.

The first LSA is always generated immediately upon an OSPF topology change, and the next LSA generated is controlled by a configured minimum start interval. The subsequent LSAs generated for the same LSA are rate limited until the configured maximum interval is reached. The same LSA is defined as an LSA instance that contains the same LSA ID number, LSA type, and advertising router ID.

If an instance of the same LSA arrives sooner than the configured receive interval, the LSA is dropped.

Note We recommend that you use an arrival interval that is less than or equal to the hold-time interval.

Components of MPLS Layer 3 VPNs

An MPLS-based Layer 3 VPN network has three components:

1. VPN route target communities—A VPN route target community is a list of all members of a Layer 3 VPN community. You must configure the VPN route targets for each Layer 3 VPN community member.
2. Multiprotocol BGP peering of VPN community PE routers—Multiprotocol BGP propagates VRF reachability information to all members of a VPN community. You must configure Multiprotocol BGP peering in all PE routers within a VPN community.
3. MPLS forwarding—MPLS transports all traffic between all VPN community members across a VPN enterprise or service provider network.

A one-to-one relationship does not necessarily exist between customer sites and VPNs. A site can be a member of multiple VPNs. However, a site can associate with only one VRF. A customer-site VRF contains all the routes that are available to the site from the VPNs of which it is a member.

High Availability and ISSU for MPLS Layer 3 VPNs

The Cisco NX-OS architecture and high availability (HA) infrastructure enables feature components to restart and resume operations transparently to other services on the device and on neighboring devices. This process allows for continuous operation and minimal data loss during planned software changes and unplanned software failures.

MPLS 6PE/6VPE supports these Cisco NX-OS HA features:

- Nonstop forwarding (NSF)
- Stateful HA

MPLS 6PE/6VPE supports these Cisco NX-OS HA technologies to allow NSF and stateful HA:

- Stateful process restart
- Stateful switchover (SSO)
- In-Service Software Upgrade (ISSU)

MPLS Layer 3 VPN supports these Cisco NX-OS HA technologies:

- NSF of Layer 2 traffic
- Graceful (stateless) restart of Layer 3 processes
- SSO
- ISSU

Note NSF requires that graceful restart is enabled in BGP and LDP.

BGP has graceful restart extensions for labels that are received from peers and recovers the local labels that are allocated for VPN routes across a BGP restart or for a supervisor switchover. BGP does not support stateful restart but on a supervisor switchover, BGP does a stateless recovery through graceful restart procedures. Cisco NX-OS forces a supervisor switchover if the BGP process fails to restart after two attempts.

The PE-CE protocols are either stateful or use graceful restart for routes that are learned from locally connected CEs. The forwarding plane continues to switch packets both for IPv4 and IPv6 routes as well as MPLS labels during any component restart or supervisor switchover.

Hub-and-Spoke Topology

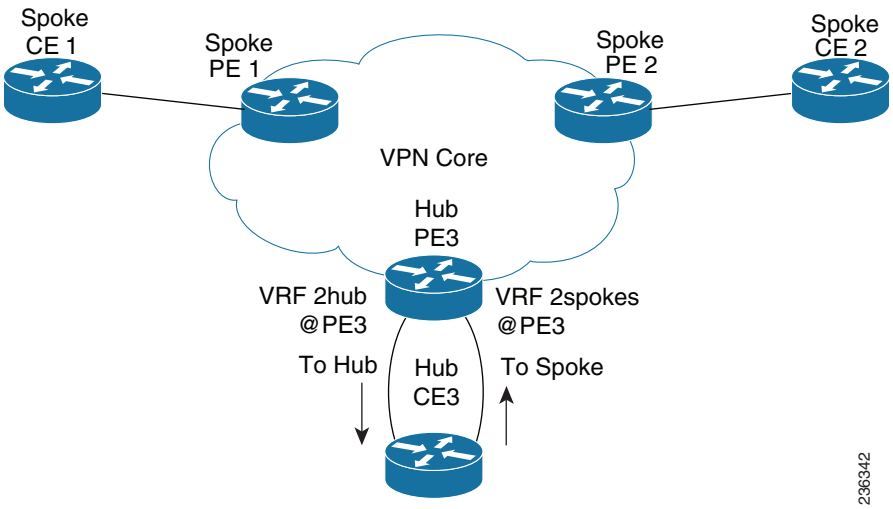
A hub-and-spoke topology prevents local connectivity between subscribers at the spoke provider edge (PE) routers and ensures that a hub site provides subscriber connectivity. Any sites that connect to the same PE router must forward intersite traffic using the hub site. This topology ensures that the routing at the spoke sites moves from the access-side interface to the network-side interface or from the network-side interface to the access-side interface but never from the access-side interface to the access-side interface. A hub-and-spoke topology allows you to maintain access restrictions between sites.

A hub-and-spoke topology prevents situations where the PE router locally switches the spokes without passing the traffic through the hub site. This topology prevents subscribers from directly connecting to each other.

A hub-and-spoke topology does not require one VRF for each spoke.

Figure 20-2 shows a sample hub-and-spoke topology.

Figure 20-2 Hub-and-Spoke Topology



As shown in the figure, a hub-and-spoke topology is typically set up with a hub PE that is configured with two VRFs:

- VRF 2hub with a dedicated link connected to the hub customer edge (CE).
- VRF 2spokes with another dedicated link connected to the hub CE.

Interior Gateway Protocol (IGP) or external BGP (eBGP) sessions are usually set up through the hub PE-CE links. The VRF 2hub imports all the exported route targets from all the spoke PEs. The hub CE learns all routes from the spoke sites and readvertises them back to the VRF 2spoke of the hub PE. The VRF 2spoke exports all these routes to the spoke PEs.

If you use eBGP between the hub PE and hub CE, you must allow duplicate autonomous system (AS) numbers in the path which is normally prohibited. You can configure the router to allow this duplicate AS number at the neighbor of VRF 2spokes of the hub PE and also for VPN address family neighbors at all the spoke PEs. In addition, you must disable the peer AS number check at the hub CE when distributing routes to the neighbor at VRF 2spokes of the hub PE.

Reverse Path Forwarding Check

The unicast Reverse Path Forwarding (uRPF) check ensures that an IP packet that enters a router uses the correct inbound interface. A hub-and-spoke configuration supports uRPF checks on the spoke-side interfaces. Because different virtual routing and forwarding instances (VRFs) are used for downstream and upstream forwarding, the uRPF mechanism ensures that source address checks occur in the downstream VRF.

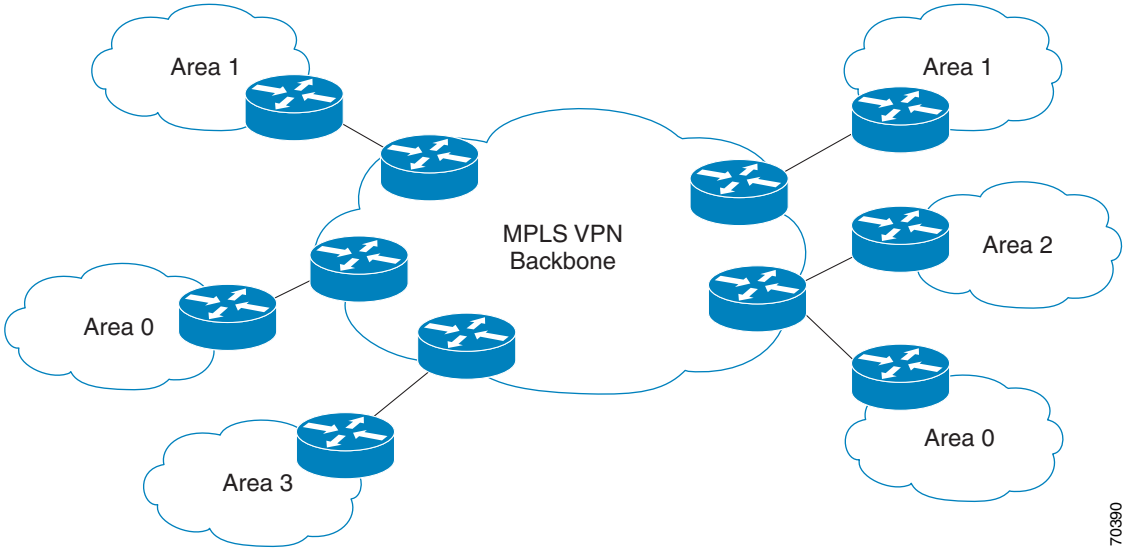
OSPF Sham-Link Support for MPLS VPN

In a Multiprotocol Label Switching (MPLS) VPN configuration, you can use the Open Shortest Path First (OSPF) protocol to connect customer edge (CE) devices to service provider edge (PE) devices in the VPN backbone. Many customers run OSPF as their intrasite routing protocol, subscribe to a VPN service, and want to exchange routing information between their sites using OSPF (during migration or on a permanent basis) over an MPLS VPN backbone.

The benefits of the OSPF sham-link support for MPLS VPN are as follows:

- Client site connection across the MPLS VPN Backbone—A sham link ensures that OSPF client sites that share a backdoor link can communicate over the MPLS VPN backbone and participate in VPN services.
- Flexible routing in an MPLS VPN configuration—In an MPLS VPN configuration, the OSPF cost tha is configured with a sham link allows you to decide if OSPF client site traffic is routed over a backdoor link or through the VPN backbone.

The figure below shows an example of how VPN client sites that run OSPF can connect over an MPLS VPN backbone.

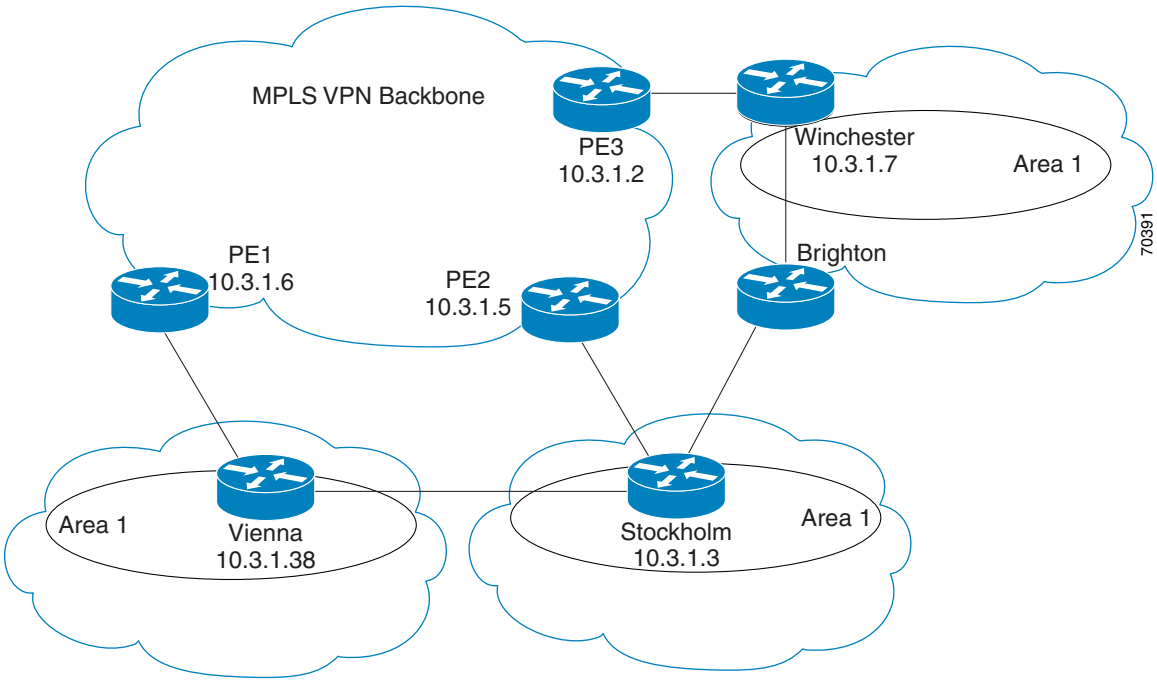


When you use OSPF to connect PE and CE devices, all routing information learned from a VPN site is placed in the VPN routing and forwarding (VRF) instance that is associated with the incoming interface. The PE devices that attach to the VPN use the Border Gateway Protocol (BGP) to distribute VPN routes to each other. A CE device can learn the routes to other sites in the VPN by peering with its attached PE device. The MPLS VPN super backbone provides an additional level of routing hierarchy to interconnect the VPN sites that are running OSPF.

When OSPF routes are propagated over the MPLS VPN backbone, additional information about the prefix in the form of BGP extended communities (route type, domain ID extended communities) is appended to the BGP update. This community information is used by the receiving PE device to decide the type of link-state advertisement (LSA) to be generated when the BGP route is redistributed to the OSPF PE-CE process. In this way, internal OSPF routes that belong to the same VPN and are advertised over the VPN backbone are seen as interarea routes on the remote sites.

Correcting OSPF Backdoor Routing

Although the Open Shortest Path First (OSPF) provider edge-to-customer edge (PE-CE) connections assume that the only path between two client sites is across the Multiprotocol Layer Switching (MPLS) VPN backbone, backdoor paths between VPN sites (shown in gray in the figure below) might exist. If these sites belong to the same OSPF area, the device chooses a path over a backdoor link because OSPF prefers intra-area paths to interarea paths. (PE devices advertise OSPF routes learned over the VPN backbone as interarea paths.) Therefore, routing is performed based on policy.



For example, the figure above shows three client sites, each with backdoor links. Because each site runs OSPF within the same Area 1 configuration, all routing between the three sites follows the intra-area path across the backdoor links, rather than over the MPLS VPN backbone.

The following example shows Border Gateway Protocol (BGP) routing table entries for the prefix 10.3.1.7/32 in the PE-1 device in the figure above. This prefix is the loopback interface of the Winchester CE device. As shown in bold in this example, the loopback interface is learned through BGP from PE-2 and PE-3. It is also generated through redistribution into BGP on PE-1.

```
PE-1# show ip bgp vpnv4 all 10.3.1.7
BGP routing table entry for 100:251:10.3.1.7/32, version 58
Paths: (3 available, best #2)
  Advertised to non peer-group peers:
  10.3.1.2 10.3.1.5
  Local
   10.3.1.5 (metric 30) from 10.3.1.5 (10.3.1.5)
    Origin incomplete, metric 22, localpref 100, valid, internal
    Extended Community: RT:1:793 OSPF DOMAIN ID:0.0.0.100 OSPF
    RT:1:2:0 OSPF 2
Local
  10.2.1.38 from 0.0.0.0 (10.3.1.6)
  Origin incomplete, metric 86, localpref 100, weight 32768,
  valid, sourced, best
  Extended Community: RT:1:793 OSPF DOMAIN ID:0.0.0.100 OSPF
  RT:1:2:0 OSPF 2
Local
  10.3.1.2 (metric 30) from 10.3.1.2 (10.3.1.2)
   Origin incomplete, metric 11, localpref 100, valid, internal
   Extended Community: RT:1:793 OSPF DOMAIN ID:0.0.0.100 OSPF
   RT:1:2:0 OSPF 2
```

Within BGP, the locally generated route (10.2.1.38) is considered to be the best route. However, as shown in bold in the next example, the VRF routing table shows that the selected path is learned through OSPF with a next hop of 10.2.1.38, shown in the figure as the Vienna CE device.

```
PE-1# show ip route vrf ospf 10.3.1.7
Routing entry for 10.3.1.7/32
  Known via "ospf 100", distance 110, metric 86, type intra area
  Redistributing via bgp 215
  Advertised by bgp 215
  Last update from 10.2.1.38 on Serial0/0/0, 00:00:17 ago
  Routing Descriptor Blocks:
  * 10.2.1.38
, from 10.3.1.7, 00:00:17 ago, via Serial0/0/0
    Route metric is 86, traffic share count is 1
```

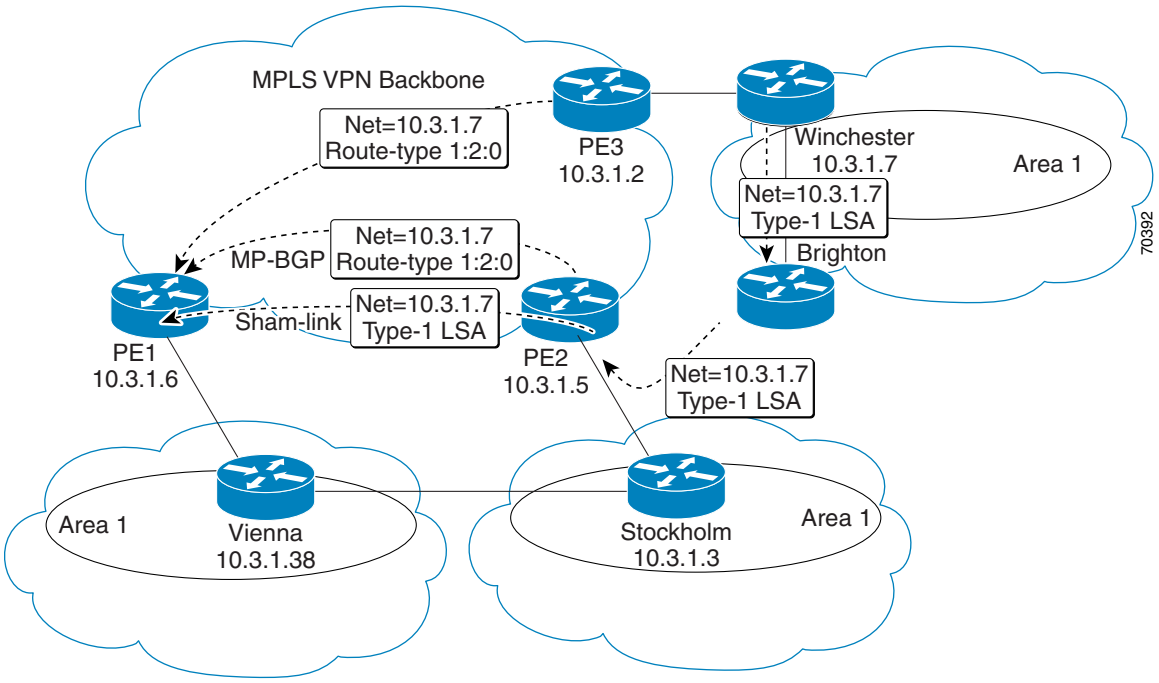
This path is selected for the following reasons:

- The OSPF intra-area path is preferred over the interarea path (over the MPLS VPN backbone) that is generated by the PE-1 device.
- OSPF has a lower administrative distance (AD) than internal BGP (BGP running between devices in the same autonomous system).

If the backdoor links between sites are used only for backup and do not participate in the VPN service, the default route selection shown in the preceding example is not acceptable. To reestablish the desired path selection over the MPLS VPN backbone, you must create an additional OSPF intra-area (logical) link between ingress and egress VRFs on the relevant PE devices. This link is called a sham link.

A sham link is required between any two VPN sites that belong to the same OSPF area and share an OSPF backdoor link. If no backdoor link exists between the sites, no sham link is required.

The figure below shows a sample sham link between PE-1 and PE-2. A cost is configured with each sham link and is used to decide whether traffic is sent over the backdoor path or the sham-link path. When a sham link is configured between PE devices, the PEs can populate the VRF routing table with the OSPF routes learned over the sham link.



Because the sham link is seen as an intra-area link between PE devices, an OSPF adjacency is created and database exchange (for the particular OSPF process) occurs across the link. The PE device can then flood LSAs between sites from across the MPLS VPN backbone. As a result, intra-area connectivity is created.

Licensing Requirements for MPLS Layer 3 VPNs

Product	License Requirement
Cisco NX-OS	MPLS Layer 3 VPNs require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.* **Note** VRF lite does not require an MPLS license for route leaking.

Prerequisites for MPLS Layer 3 VPNs

MPLS Layer 3 VPNs has the following prerequisites:

- Ensure that you have configured MPLS and Label Distribution Protocol (LDP) or Resource Reservation Protocol (RSVP) Traffic Engineering (TE) in your network. All routers in the core, including the PE routers, must be able to support MPLS forwarding.
- Ensure that you have installed the correct license for MPLS and any other features you will be using with MPLS.

Guidelines and Limitations for MPLS Layer 3 VPNs

MPLS Layer 3 VPNs have the following configuration guidelines and limitations:

- MPLS Layer 3 VPNs support the following CE-PE routing protocols:
  - BGP (IPv4 and IPv6)
  - Enhanced Interior Gateway Protocol (EIGRP) (IPv4)
  - Open Shortest Path First (OSPFv2)
  - Routing Information Protocol (RIPv2)

Note Cisco NX-OS supports static routes (IPv4 and IPv6) for PE-CE routing.

- Set statements in an import route map are ignored.
- The BGP minimum route advertisement interval (MRAI) value for all iBGP and eBGP sessions is zero and is not configurable.
- In a high scale setup with many BGP routes getting redistributed into EIGRP, modify the EIGRP signal timer to ensure that the EIGRP convergence time is higher than the BGP convergence time. This process allows all the BGP routes to be redistributed into EIGRP, before EIGRP signals convergence.
- For Cisco NX-OS releases before Cisco NX-OS Release 5.2(5), the EIGRP site of origin requires an MPLS license and he MPLS Layer 3 VPN feature is enabled. Beginning with Cisco NX-OS Release 5.2(5), the EIGRP site of origin feature does not require an MPLS license.
- Beginning with Cisco Nx-OS Release 7.3(0)DX(1), MPLS Layer 3 VPNs are supported on M3 Series modules.

OSPF sham-link support for MPLS VPN has the following guideline and limitation:

- When OSPF is used as a protocol between PE and CE devices, the OSPF metric is preserved when routes are advertised over the VPN backbone. The metric is used on the remote PE devices to select the correct route. Do not modify the metric value when OSPF is redistributed to BGP and when BGP is redistributed to OSPF. If you modify the metric value, routing loops might occur.

Default Settings for MPLS Layer 3 VPNs

Table 20-1 lists the default settings for MPLS Layer 3 VPN parameters.

Table 20-1 Default MPLS Layer 3 VPN Parameters

Parameters	Default
L3VPN feature	Disabled
L3VPN SNMP notifications	Disabled
allowas-in (for a hub-and-spoke topology)	0
disable-peer-as-check (for a hub-and-spoke topology)	Disabled

Configuring MPLS Layer 3 VPNs

This section includes the following topics:

Configuring the Core Network

This section includes the following topics:

Assessing the Needs of MPLS Layer 3 VPN Customers

You can identify the core network topology so that it can best serve MPLS Layer 3 VPN customers.

Step 1 Identify the size of the network:

- Identify the following to determine the number of routers and ports you need:
- How many customers do you need to support?
- How many VPNs are needed per customer?
- How many virtual routing and forwarding instances are there for each VPN?

Step 2 Determine which routing protocols you need in the core network.

Step 3 Determine if you need MPLS VPN high availability support.

Note MPLS VPN nonstop forwarding and graceful restart are supported on select routers and Cisco NX-OS releases. You need to make sure that graceful restart for BGP and LDP is enabled.

Step 4 Configure the routing protocols in the core network.

Note See the *Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide* for configuration steps.

Step 5 Determine if you need BGP load sharing and redundant paths in the MPLS Layer 3 VPN core.

Note See the "Configuring MPLS Layer 3 VPN Load Balancing" section on page 22-405 for more information.

Configuring MPLS in the Core

To enable MPLS on all routers in the core, you must configure a label distribution protocol. You can use either of the following as a label distribution protocol:

- MPLS Label Distribution Protocol (LDP).
- MPLS Traffic Engineering Resource Reservation Protocol (RSVP).

Configuring Multiprotocol BGP on the PE Routers and Route Reflectors

You can configure multiprotocol BGP connectivity on the PE routers and route reflectors.

Prerequisites

- Ensure that you are in the correct virtual device context (VDC) (or use the **switchto vdc** command).
- Ensure that graceful restart is enabled on all routers for BGP and LDP.

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 5	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.255`	(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

	Command	Purpose
Step 7	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 209.165.201.1 remote-as 1.1` `switch(config-router-neighbor)#`	Adds an entry to the iBGP neighbor table. The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation.
Step 8	**address-family** {**vpnv4** | **vpnv6**} **unicast** **Example:** `switch(config-router-neighbor)# address-family vpnv4 unicast` `switch(config-router-neighbor-af)#`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard VPNv4 or VPNv6 address prefixes.
Step 9	**send-community extended** **Example:** `switch(config-router-neighbor-af)# send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor.
Step 10	**show bgp** {**vpnv4** | **vpnv6**} **unicast neighbors** **Example:** `switch(config-router-neighbor-af)# show bgp vpnv4 unicast neighbors`	(Optional) Displays information about BGP neighbors.
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Connecting the MPLS VPN Customers

This section includes the following topics:

Defining VRFs on the PE Routers to Enable Customer Connectivity

You must create VRFs on the PE routers to enable customer connectivity. You configure route targets to control which IP prefixes are imported into the customer VPN site and which IP prefixes are exported to the BGP network. You can optionally use an import or export route map to provide more fine-grained control over the IP prefixes that are imported into the customer VPN site or exported out of the VPN site. You can use a route map to filter routes that are eligible for import or export in a VRF, based on the route target extended community attributes of the route. The route map might, for example, deny access to selected routes from a community that is on the import route target list.

Note If you are using import maps, you must configure an import statement in order for the import map to take effect. Similarly, you must configure an export statement in order for the export map to take effect. Beginning with Cisco NX-OS Release 5.2(5), however, an export statement is not required in order for the export map to take effect.

Note Beginning with Cisco NX-OS Releases 5.2(7) and 6.1(2), import maps support matching and setting on standard and extended communities. In earlier releases, import maps do not support matching and setting on standard and extended communities. Beginning with Cisco NX-OS Release 5.2(1), export maps support matching and setting on standard and extended communities.

Note Do not use the **export map** command in the VRF mode for prefix filtering. When a route-target export is configured, all routes are exported and then imported to VRFs with a matching route-target import. In this case, the export map does not filter routes, but it can be used to set attributes for the selected routes. If you need to export only the selected routes, remove the route-target export and use the export map to filter routes; and set the extcommunity rt xx:xx so that the VRFs with the matching route-target import imports these routes.

Note Using the **continue** *sequence-number* keyword under the under VRF export/import map in the route-map configuration mode is not supported.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **vrf context** *vrf-name*
5. **rd** *route-distinguisher*
6. **address-family** {**ipv4** | **ipv6**} **unicast**
7. **route-target** {**import** | **export**} *route-target-ext-community*
8. (Optional) **maximum routes** *max-prefix [***threshold** *value] [***reinstall]**
9. (Optional) **import** [**vrf default** *max-prefix*] **map** *route-map*
10. (Optional) **show vrf** *vrf-name*
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 4	**vrf context** *vrf-name* **Example:** `switch(config)# vrf context vpn1` `switch(config-vrf)#`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd 1.2:1`	Configures the route distinguisher. The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter an RD in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 6	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast` `switch(config-vrf-af-ipv4)#`	Specifies the IPv4 address family type and enters address family configuration mode.

	Command	Purpose
Step 7	**route-target** {**import** |**export**} *route-target-ext-community* **Example:** `switch(config-vrf-af-ipv4)# route-target import 1.0:1`	Specifies a route-target extended community for a VRF as follows: • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the *route-target-ext-community* argument in either of these formats: – 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 – 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 8	**maximum routes** *max-routes* [**threshold** *value*] [**reinstall**] **Example:** `switch(config-vrf-af-ipv4)# maximum routes 10000`	(Optional) Configures the maximum number of routes that can be stored in the VRF route table. The *max-routes* range is from 1 to 4294967295. The threshold value range is from 1 to 100.
Step 9	**import** [**vrf default** *max-prefix*] **map** *route-map* **Example:** `switch(config-vrf-af-ipv4)# import vrf default map vpn1-route-map`	(Optional) Configures an import policy for a VRF to import prefixes from the default VRF as follows: • The *max-prefix* range is from 1 to 2147483647. The default is 1000 prefixes. • The *route-map* argument specifies the route map to be used as an import route map for the VRF and can be any case-sensitive, alphanumeric string up to 63 characters.
Step 10	**show vrf** *vrf-name* **Example:** `switch(config-vrf-af-ipv4)# show vrf vpn1`	(Optional) Displays information about a VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 11	**copy running-config startup-config** **Example:** `switch(config-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring VRF Interfaces on PE Routers for Each VPN Customer

You can associate a virtual routing and forwarding instance (VRF) with an interface or subinterface on the PE routers.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*
3. **vrf member** *vrf-name*
4. (Optional) **show vrf** *vrf-name* **interface**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `switch(config)# interface Ethernet 5/0` `switch(config-if)#`	Specifies the interface to configure and enters interface configuration mode as follows: • The *type* argument specifies the type of interface to be configured. • The *number* argument specifies the port, connector, or interface card number.
Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member vpn1`	Associates a VRF with the specified interface or subinterface. The *vrf-name* argument is the name assigned to a VRF.
Step 4	**show vrf** *vrf-name* **interface** **Example:** `switch(config-if)# show vrf vpn1 interface`	(Optional) Displays information about interfaces associated with a VRF. The *vrf-name* argument is any case-sensitive alphanumeric string up to 32 characters.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring Routing Protocols Between the PE and CE Routers

This section includes the following topics:

Configuring Static or Directly Connected Routes Between the PE and CE Routers

You can configure the PE router for PE-to-CE routing sessions that use static routes.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. {**ip** | **ipv6**} **route** *prefix/mask nexthop*
4. **address-family** {**ipv4** | **ipv6**} **unicast**
5. **feature bgp**
6. **router bgp** *as-number*
7. **vrf** *vrf-name*
8. **address-family** {**ipv4** | **ipv6**} **unicast**
9. **redistribute static route-map** *map-tag*
10. **redistribute direct route-map** *map-tag*
11. (Optional) **show** {**ipv4** | **ipv6**} **route static vrf** *vrf-name*
12. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**vrf context** *vrf-name* **Example:** `switch(config)# vrf context vpn1` `switch(config-vrf)#`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 3	{**ip** | **ipv6**} **route** *prefix nexthop* **Example:** `switch(config-vrf)# ip route 192.0.2.1/28 ethernet 2/1`	Defines static route parameters for every PE-to-CE session. The *prefix* and *nexthop* are as follows: • IPv4—in dotted decimal notation • IPv6—in hex format.
Step 4	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast` `switch(config-vrf-af)#`	Specifies the IPv4 address family type and enters address family configuration mode.

	Command	Purpose
Step 5	**feature bgp** **Example:** `switch(config-vrf-af)# feature bgp` `switch(config)#`	Enables the BGP feature.
Step 6	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 7	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config--router-vrf)#`	Associates the BGP process with a VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 8	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-vrf)# address-family ipv4 unicast` `switch(config-router-vrf-af)#`	Specifies the address family type and enters address family configuration mode.
Step 9	**redistribute static route-map** *map-name* **Example:** `switch(config-router-vrf-af)# redistribute static route-map StaticMap`	Redistributes static routes into BGP. The *map-name* can be any case-sensitive, alphanumeric string up to 63 characters.
Step 10	**redistribute direct route-map** *map-name* **Example:** `switch(config-router-vrf-af)# redistribute direct route-map DirectMap`	Redistributes directly connected routes into BGP. The *map-name* can be any case-sensitive, alphanumeric string up to 63 characters.
Step 11	**show** {**ipv4** | **ipv6**} **route vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show ip ipv4 route vrf vpn1`	(Optional) Displays information about routes. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 12	**copy running-config startup-config** **Example:** `switch(config-router-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring BGP as the Routing Protocol Between the PE and CE Routers

You can use eBGP to configure the PE router for PE-to-CE routing sessions.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **router bgp** *as-number*
4. **vrf** *vrf-name*
5. **neighbor** *ip-address* **remote-as** *as-number*
6. **address-family** {**ipv4** | **ipv6**} **unicast**
7. **show bgp** {**ipv4** | **ipv6**} **unicast neighbors vrf** *vrf-name*
8. **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp` `switch(config)#`	Enables the BGP feature.
Step 3	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 4	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config--router-vrf)#`	Associates the BGP process with a VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 1.2` `switch(config-router-vrf-neighbor)#`	Adds an entry to the eBGP neighbor table. • The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.

	Command	Purpose
Step 6	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-vrf-neighbor)# address-family ipv4 unicast` `switch(config-router-vrf-neighbor-af)#`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard IPv4 or IPv6 address prefixes.
Step 7	**show bgp** {**ipv4** | **ipv6**} **unicast neighbors vrf** *vrf-name* **Example:** `switch(config-router--vrf-neighbor-af)# show bgp ipv4 unicast neighbors vrf vpn1`	(Optional) Displays information about BGP neighbors. The *vrf-name* argument is any case-sensitive alphanumeric string up to 32 characters.
Step 8	**copy running-config startup-config** **Example:** `switch(config-router-vrf-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring RIPv2 Between the PE and CE Routers

You can use RIP to configure the PE router for PE-to-CE routing sessions.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature rip**
3. **router rip** *instance-tag*
4. **vrf** *vrf-name*
5. **address-family ipv4 unicast**
6. **redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
7. (Optional) **show ip rip vrf** *vrf-name*
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature rip** **Example:** `switch(config)# feature rip`	Enables the RIP feature.

	Command	Purpose
Step 3	**router rip** *instance-tag* **Example:** `switch(config)# router rip Test1` `switch(config-router)#`	Enables RIP and enters router configuration mode. The *instance-tag* can be any case-sensitive, alphanumeric string up to 20 characters.
Step 4	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config--router-vrf)#`	Associates the RIP process with a VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**address-family ipv4 unicast** **Example:** `switch(config-router-vrf)# address-family ipv4 unicast` `switch(config-router-vrf-af)#`	Specifies the address family type and enters address family configuration mode.
Step 6	**redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example:** `switch(config-router-vrf-af)#` `redistribute bgp 1.0 route-map bagpipe`	Redistributes routes from one routing domain into another routing domain. The *as* number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. The *instance-tag* can be any case-sensitive alphanumeric string up to 20 characters.
Step 7	**show ip rip vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show ip rip vrf vpn1`	(Optional) Displays information about RIP. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 8	**copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring OSPF Between the PE and CE Routers

You can use OSPFv2 to configure the PE router for PE-to-CE routing sessions. You can optionally create an OSPF sham link if you have OSPF back door links that are not part of the MPLS network.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature ospf**
3. **router ospf** *instance-tag*
4. **vrf** *vrf-name*
5. (Optional) **area** *area-id* **sham-link** *source-address destination-address*
6. **address-family** {**ipv4** | **ipv6**} **unicast**

7. **redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
8. (Optional) **show ip ospf** *instance-tag* **vrf** *vrf-name*
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature ospf** **Example:** `switch(config)# feature ospf`	Enables the OSPF feature.
Step 3	**router ospf** *instance-tag* **Example:** `switch(config)# router ospf Test1` `switch(config-router)#`	Enables OSPF and enters router configuration mode. The *instance-tag* can be any case-sensitive, alphanumeric string up to 20 characters.
Step 4	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config-router-vrf)#`	Enters router VRF configuration mode. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**area** *area-id* **sham-link** *source-address destination-address* **Example:** `switch(config-router-vrf)# area 1 sham-link 10.2.1.1 10.2.1.2`	(Optional) Configures the sham link on the PE interface within a specified OSPF area and with the loopback interfaces specified by the IP addresses as endpoints. You must configure the sham link at both PE endpoints.
Step 6	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router)# address-family ipv4 unicast` `switch(config-router-vrf-af)#`	Specifies the address family type and enters address family configuration mode.
Step 7	**redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example:** `switch(config-router-vrf-af)# redistribute bgp 1.0 route-map bgpMap`	Redistributes routes from one routing domain into another routing domain. • The *as* number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. • The *instance-tag* can be any case-sensitive alphanumeric string up to 20 characters.

	Command	Purpose
Step 8	**show ip ospf** *instance-tag* **vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show ip ospf Test1 vrf vpn1`	(Optional) Displays information about OSPF.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring EIGRP Between the PE and CE Routers

You can configure the PE router to use Enhanced Interior Gateway Routing Protocol (EIGRP) between the PE and CE routers to transparently connect EIGRP customer networks through an MPLS-enabled BGP core network so that EIGRP routes are redistributed through the VPN across the BGP network as internal BGP (iBGP) routes.

Prerequisites

You must configure BGP in the network core.

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature eigrp**
3. **router eigrp** *instance-tag*
4. **vrf** *vrf-name*
5. **(Optional) address-family ipv4 unicast**
6. **redistribute bgp** *as-number* **route-map** *map-name*
7. **(Optional) autonomous-system** *as-number*
8. (Optional) **show ipv4 eigrp vrf** *vrf-name*
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal switch(config)#`	Enters global configuration mode.
Step 2	**feature eigrp** **Example:** `switch(config)# feature eigrp`	Enables the BGP feature.

	Command	Purpose
Step 3	**router eigrp** *instance-tag* **Example:** `switch(config)# router eigrp Test1` `switch(config-router)#`	Configures an EIGRP instance and enters router configuration mode. The *instance-tag* can be any case-sensitive, alphanumeric string up to 20 characters.
Step 4	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config-router-vrf)#`	Enters router VRF configuration mode. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**address-family ipv4 unicast** **Example:** `switch(config-router-vrf)#` `address-family ipv4 unicast` `switch(config-router-vrf-af)#`	(Optional) Enters address family configuration mode for configuring routing sessions that use standard IPv4 address prefixes.
Step 6	**redistribute bgp** *as-number* **route-map** *map-name* **Example:** `switch(config-router-vrf-af)# redis` `tribute bgp 1.0 route-map BGPMap`	Redistributes BGP into the EIGRP. • The autonomous system number of the BGP network is configured in this step. BGP must be redistributed into EIGRP for the CE site to accept the BGP routes that carry the EIGRP information. A metric must also be specified for the BGP network. • The *map-name* can be any case-sensitive, alphanumeric string up to 63 characters.
Step 7	**autonomous-system** *as-number* **Example:** `switch(config-router-vrf-af)#` `autonomous-system 1.3`	(Optional) Specifies the autonomous system number for this address family for the customer site. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 8	**show ip eigrp vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show` `ipv4 eigrp vrf vpn1`	(Optional) Displays information about EIGRP in this VRF. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-vrf-af)# copy` `running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring PE-CE Redistribution in BGP for the MPLS VPN

You must configure BGP to distribute the PE-CE routing protocol on every PE router that provides MPLS Layer 3 VPN services if the PE-CE protocol is not BGP.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Restrictions

Redistribution between native EIGRP VRFs is not supported.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **router bgp** *as-number*
4. (Optional) **router-id** *ip-address*
5. **neighbor** *ip-address* **remote-as** *as-number*
6. **update-source loopback** [**0** | **1**]
7. **address-family** {**vpnv4** | **vpnv6**}
8. **send-community extended**
9. **vrf** *vrf-name*
10. **address-family** {**ipv4** | **ipv6**} **unicast**
11. **redistribute** {**direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
12. (Optional) **show bgp** {**ipv4** | **ipv6**} **unicast vrf** *vrf-name*
13. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 4	**router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.255`	(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

	Command	Purpose
Step 5	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 209.165.201.1 remote-as 1.2` `switch(config-router-neighbor)#`	Adds an entry to the BGP or multiprotocol BGP neighbor table.The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation.The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 6	**update-source loopback** [**0** | **1**] **Example:** `switch(config-router-neighbor)# update-source loopback 0#`	Specifies the source address of the BGP session.
Step 7	**address-family** {**vpnv4** | **vpnv6**}[**unicast**] **Example:** `switch(config-router-neighbor)# address-family vpnv4` `switch(config-router-neighbor-af)#`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard VPNv4 or VPNv6 address prefixes. The optional **unicast** keyword specifies VPNv4 or VPNv6 unicast address prefixes.
Step 8	**send-community extended** **Example:** `switch(config-router-neighbor-af)# send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor.
Step 9	**vrf** *vrf-name* **Example:** `switch(config-router-neighbor-af)# vrf vpn1` `switch(config-router-vrf)#`	Enters router VRF configuration mode. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 10	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-vrf)# address-family ipv4 unicast` `switch(config-router-vrf-af)#`	Enters address family configuration mode for configuring routing sessions that use standard IPv4 or IPv6 address prefixes.
Step 11	**redistribute** {**direct** | {**eigrp**| **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example:** `switch(config-router-af-vrf)# redistribute eigrp Test2 route-map EigrpMap`	Redistributes routes from one routing domain into another routing domain. The *as* number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. The *instance-tag* can be any case-sensitive, alphanumeric string up to 20 characters. The *map-name* can be any case-sensitive alphanumeric string up to 63 characters.
Step 12	**show bgp** {**ipv4** | **ipv6**} **unicast vrf** *vrf-name* **Example:** `switch(config-router--vrf-af)# show bgp ipv4 unicast vrf vpn1`	(Optional) Displays information about BGP. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 13	**copy running-config startup-config** **Example:** `switch(config-router-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring a Hub-and-Spoke Topology

This section includes the following topics:

Configuring VRFs on the Hub PE Router

You can configure hub and spoke VRFs on the hub PE router.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 4	**vrf context** *vrf-hub* **Example:** `switch(config)# vrf context 2hub` `switch(config-vrf)#`	Defines the VPN routing instance for the PE hub by assigning a VRF name and enters VRF configuration mode. The *vrf-hub* argument is any case-sensitive alphanumeric string up to 32 characters.
Step 5	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd 1:103`	Creates routing and forwarding tables. The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter a route distinguisher in either of these formats: – 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 – 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1

	Command	Purpose
Step 6	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast` `switch(config-vrf-af-ipv4)#`	Specifies the IPv4 address family type and enters address family configuration mode.
Step 7	**route-target** {**import** |**export**} *route-target-ext-community* **Example:** `switch(config-vrf-af-ipv4)# route-target import 1:101`	Creates a route-target extended community for a VRF. • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the *route-target-ext-community* argument as follows: – 16-bit or 32-bit AS number, such as your 32-bit number, 1.2:3 – 32-bit IP address, such as your 16-bit number, 192.0.2.1:1
Step 8	**vrf context** *vrf-spoke* **Example:** `switch(config-vrf-af-ipv4)# vrf context 2spokes` `switch(config-vrf)#`	Defines the VPN routing instance for the PE spoke by assigning a VRF name and enters VRF configuration mode. The *vrf-spoke* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 9	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast` `switch(config-vrf-af-ipv4)#`	Specifies the IPv4 address family type and enters address family configuration mode.
Step 10	**route-target** {**import** |**export**} *route-target-ext-community* **Example:** `switch(config-vrf-af-ipv4)# route-target export 1:100`	Creates a route-target extended community for a VRF. The **import** keyword imports routing information from the target VPN extended community. The **export** keyword exports routing information to the target VPN extended community. The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the *route-target-ext-community* argument in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1

	Command	Purpose
Step 11	**show running-config vrf** *vrf-name* **Example:** `switch(config-vrf-af-ipv4)# show running-config vrf 2spokes`	(Optional) Displays the running configuration for the VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 12	**copy running-config startup-config** **Example:** `switch(config-vrf-af-ipv4)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring eBGP on the Hub PE Router

You can use eBGP to configure PE-to-CE hub routing sessions.

Note If all CE sites are using the same BGP AS number, you must perform the following tasks:

- Configure either the BGP **as-override** command at the PE (hub) **or the allowas-in** command at the receiving CE router.
- To advertise BGP routes learned from one ASN back to the same ASN, configure the **disable-peer-as-check** command at the PE router to prevent loopback.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 4	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.

	Command	Purpose
Step 5	**router bgp** *as-number* **Example:** switch(config)# router bgp 1.1 switch(config-router)#	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**neighbor** *ip-address* **remote-as** *as-number* **Example:** switch(config-router)# neighbor 209.165.201.1 remote-as 1.2 switch(config-router-neighbor)#	Adds an entry to the BGP or multiprotocol BGP neighbor table. • The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 7	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** switch(config-router-vrf-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	Specifies the IP address family type and enters address family configuration mode.
Step 8	**send-community extended** **Example:** switch(config-router-neighbor-af)# send-community extended	(Optional) Configures BGP to advertise extended community lists.
Step 9	**vrf** *vrf-hub* **Example:** switch(config-router-neighbor-af)# vrf 2hub switch(config-router-vrf)#	Enters VRF configuration mode. The *vrf-hub* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 10	**neighbor** *ip-address* **remote-as** *as-number* **Example:** switch(config-router-vrf)# neighbor 33.0.0.33 1 remote-as 150 switch(config-router-vrf-neighbor)#	Adds an entry to the BGP or multiprotocol BGP neighbor table for this VRF. • The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 11	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** switch(config-router-vrf-neighbor)# address-family ipv4 unicast switch(config-router--vrf-neighbor-af)#	Specifies the IP address family type and enters address family configuration mode.

	Command	Purpose
Step 12	**as-override** **Example:** `switch(config-router-vrf-neighbor-af)# as-override`	(Optional) Overrides the AS-number when sending an update. If all BGP sites are using the same AS number, coof the cfollowing commands: • Configure the BGP **as-override** command at the PE (hub) **or** • **Configure the allowas-in** command at the receiving CE router.
Step 13	**vrf** *vrf-spoke* **Example:** `switch(config-router-vrf-neighbor-af)# vrf 2spokes` `switch(config-router-vrf)#`	Enters VRF configuration mode. The *vrf-spoke* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 14	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router-vrf)# neighbor 33.0.1.33 1 remote-as 150` `switch(config-router-vrf-neighbor)#`	Adds an entry to the BGP or multiprotocol BGP neighbor table for this VRF. • The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 15	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router--vrf-neighbor)# address-family ipv4 unicast` `switch(config-router-vrf-neighbor-af)#`	Specifies the IPv4 address family type and enters address family configuration mode.
Step 16	**allowas-in** [*number*] **Example:** `switch(config-router-vrf-neighbor-af)# allowas-in 3`	(Optional) Allows duplicate AS numbers in the AS path. Configure this parameter in the VPN address family configuration mode at the PE spokes and at the neighbor mode at the PE hub.
Step 17	**show running-config bgp** **Example:** `switch(config-router-vrf-neighbor-af)# show running-config bgp`	(Optional) Displays the running configuration for BGP.
Step 18	**copy running-config startup-config** **Example:** `switch(config-router-vrf-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring eBGP on the Hub CE Router

You can use eBGP to configure PE-to-CE hub routing sessions.

Note If all CE sites are using the same BGP AS number, you must perform the following tasks:

- Configure either the **as-override** command at the PE (hub) **or the allowas-in** command at the receiving CE router.
- Configure the **disable-peer-as-check** command at the CE router.
- To advertise BGP routes learned from one ASN back to the same ASN, configure the **disable-peer-as-check** command at the PE router to prevent loopback.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 4	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 5	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**neighbor** *hub-ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 33.0.0.63 remote-as 100` `switch(config-router-neighbor)#`	Adds an entry to the BGP or multiprotocol BGP neighbor table. • The *hub-ip-address* argument specifies the IPv4 or IPv6 address of the neighbor hub. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.

	Command	Purpose
Step 7	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-neighbor)# address-family ipv4 unicast` `switch(config-router-neighbor-af)#`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.
Step 8	**allowas-in** *number* **Example:** `switch(config-router-vrf-neighbor-af)# allowas-in 3`	(Optional) Allows an AS path with the PE ASN for a specified number of times. • The range is from 1 to 10. • If all BGP sites are using the same AS number, coof the cfollowing commands: – Configure the BGP **as-override** command at the PE (hub) **or** – **Configure the allowas-in** command at the receiving CE router.
Step 9	**neighbor** *spoke-ip-address* **remote-as** *as-number* **Example:** `switch(config-router-neighbor-af)# neighbor 33.0.1.63 remote-as 100` `switch(config-router-neighbor)#`	Adds an entry to the BGP or multiprotocol BGP neighbor table. • The *spoke-ip-address* argument specifies the IPv4 or IPv6 address of the neighbor spoke. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 10	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-neighbor)# address-family ipv4 unicast` `switch(config-router-neighbor-af)#`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.
Step 11	**disable-peer-as-check** **Example:** `switch(config-router-neighbor-af)# disable-peer-as-check`	Disables checking the peer AS number during route advertisement.
Step 12	**show running-config bgp** **Example:** `switch(config-router-neighbor-af)# show running-config bgp`	(Optional) Displays the running configuration for BGP.
Step 13	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring VRFs on the Spoke PE Router

You can configure hub and spoke VRFs on the spoke PE router.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 4	**vrf context** *vrf-spoke* **Example:** `switch(config)# vrf context spoke` `switch(config-vrf)#`	Defines the VPN routing instance for the PE spoke by assigning a VRF name and enters VRF configuration mode. The *vrf-spoke* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd 1:101`	Creates routing and forwarding tables. The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter an route distinguisher in either of the following formats: – 16-bit or 32-bit AS number, such as your 32-bit number, 1.2:3 – 32-bit IP address, such as your 16-bit number, 192.0.2.1:1
Step 6	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast` `switch(config-vrf-af-ipv4)#`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.

	Command	Purpose
Step 7	**route-target** {**import** |**export**} *route-target-ext-community* **Example:** `switch(config-vrf-af-ipv4)# route-target export 1:101`	Creates a route-target extended community for a VRF. • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the *route-target-ext-community* argument in either of the following formats: – 16-bit or 32-bit AS number, such as your 32-bit number, 1.2:3 – 32-bit IP address, such as your 16-bit number, 192.0.2.1:1
Step 8	**show running-config vrf** *vrf-name* **Example:** `switch(config-vrf-af-ipv4)# show running-config vrf 2spokes`	(Optional) Displays the running configuration for the VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 9	**copy running-config startup-config** **Example:** `switch(config-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring eBGP on the Spoke PE Router

You can use eBGP to configure PE spoke routing sessions.

Note If all CE sites are using the same BGP AS number, you must perform the following tasks:

- Configure the **the allowas-in** command at the preceiving spoke router.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS L3 VPN feature.
Step 4	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 5	**router bgp** *as-number* **Example:** `switch(config)# router bgp 100` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 63.63.0.63 remote-as 100` `switch(config-router-neighbor)#`	Adds an entry to the BGP or multiprotocol BGP neighbor table. • The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 7	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-neighbor)#` `address-family ipv4 unicast` `switch(config-router-neighbor-af)#`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.

	Command	Purpose
Step 8	**allowas-in** *number* **Example:** `switch(config-router-vrf-neighbor-af)# allowas-in 3`	(Optional) Allows an AS path with the PE ASN for a specified number of times. • The range is from 1 to 10. • If all BGP sites are using the same AS number, coof the cfollowing commands: – Configure the BGP **as-override** command at the PE (hub) **or** – **Configure the allowas-in** command at the receiving CE router.
Step 9	**send-community extended** **Example:** `switch(config-router-neighbor)# send-community extended`	(Optional) Configures BGP to advertise extended community lists.
Step 10	**show running-config bgp** **Example:** `switch(config-router-vrf-neighbor-af)# show running-config bgp`	(Optional) Displays the running configuration for BGP.
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-vrf-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Preventing Loops

You can configure the site of origin and ASN controls to prevent routing loops within a VPN.

Because CEs usually share the same ASN, to advertise BGP routes learned from one ASN back to the same ASN, the neighbor onfiguration **disable-peer-as-check** command is required. In addition, to allow BGP routes with the same ASN to be received at a CE, configure either the neighbor configuration **as-override** command or the **allowas-in** command at the PE.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example** `switch# feature bgp` `switch(config)`	Enables the BGP feature set.
Step 3	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 4	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config--router-vrf)#`	Associates the BGP process with a VRF. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 5	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 1.2` `switch(config-router-vrf-neighbor)#`	Adds an entry to the eBGP neighbor table. • The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 6	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-router-vrf-neighbor)# address-family ipv4 unicast` `switch(config-router-vrf-neighbor-af)#`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard IPv4 or IPv6 address prefixes.
Step 7	**allowas-in** *number* **Example:** `switch(config-router-vrf-neighbor-af)# allowas-in 3`	(Optional) Allows an AS path with the PE ASN for a specified number of times. The range is from 1 to 10.

	Command	Purpose
Step 8	**soo** *value* **Example:** `switch(config-router--vrf-neighbor-af)# soo 1:1`	(Optional) Configures the site of origin BGP extended community value. • The value is in one of the following formats: – asn:number – IP address:number • The number range is from 0 to 65535 for a 2-byte ASN or from 0 to 4294967295 for a 4-byte ASN.
Step 9	**as-override** **Example:** `switch(config-router--vrf-neighbor-af)# as-override`	(Optional) Configures a PE router to override the ASN of a site with the ASN of a provider.
Step 10	**show bgp** {**ipv4** | **ipv6**} **unicast neighbors vrf** *vrf-name* **Example:** `switch(config-router--vrf-neighbor-af)# show bgp ipv4 unicast neighbors vrf vpn1`	(Optional) Displays information about BGP neighbors. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-vrf-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring Sham-Link for OSPF Support of an MPLS VPN

Before You Begin

- Before you can configure a sham link in an MPLS VPN, you must enable OSPF as follows:
  - Create an OSPF routing process.
  - Specify the range of IP addresses to be associated with the routing process.
  - Assign area IDs to be associated with the range of IP addresses.
- Before you create a sham link between PE devices in an MPLS VPN, you must configure a separate /32 address on the remote PE so that OSPF packets can be sent over the VPN backbone to the remote end of the sham link. The /32 address must meet the following criteria:
  - Belong to a VRF.
  - Not be advertised by OSPF.
  - Be advertised by BGP.

Note You can use the /32 address for other sham links.

SUMMARY STEPS

1. **configure terminal**

2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature ospf**
5. **device ospf** *instance-tag*
6. **vrf** *vrf-name*
7. **area** *area-id* **sham-link** *source-address destination-address*
8. (Optional) **demand circuit**
9. **address-family** {**ipv4** | **ipv6**} **unicast**
10. **redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 4	**feature ospf** **Example:** `switch(config)# feature ospf`	Enables the OSPF feature set.
Step 5	**device ospf** *instance-tag* **Example:** `switch(config)# device ospf test1` `switch(config-device)#`	Enables OSPF and enters device configuration mode. The *instance-tag* argument is any case-sensitive, alphanumeric string up to 20 characters.
Step 6	**vrf** *vrf-name* **Example:** `switch(config-device)# vrf vpn1` `switch(config-device-vrf)#`	Enters device VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.

	Command	Purpose
Step 7	**area** *area-id* **sham-link** *source-address destination-address* **Example:** `switch(config-device-vrf)# area 1 sham-link 10.2.1.1 10.2.1.2` `switch(config-device-vrf-slink)#`	Configures the sham link on the PE interface within a specified OSPF area and with the loopback interfaces specified by the IP addresses (source and destination) as endpoints. **Note** You must configure the sham link at both PE endpoints.
Step 8	**demand circuit** **Example:** `switch(config-device-vrf-slink)# demand circuit`	(Optional) Specifies the sham link as a demand circuit (DC) by the OSPF in order to reduce the traffic flow over the sham link.
Step 9	**address-family** {**ipv4** | **ipv6**} **unicast** **Example:** `switch(config-device-vrf-slink)# address-family ipv4 unicast` `switch(config-device-vrf-af)#`	Enters address family configuration mode for configuring routing sessions, such as OSPF, that use standard IPv4 or IPv6 address prefixes.
Step 10	**redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example:** `switch(config-device-vrf-af)# redistribute bgp 1.0 route-map bgpMap`	Redistributes routes from one routing domain into another routing domain. • The *as* number is a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in the xx.xx format. • The *instance-tag* is any case-sensitive, alphanumeric string up to 20 characters.
Step 11	**copy running-config startup-config** **Example:** `switch(config-device-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS Layer 3 VPN Configuration

To display the MPLS Layer 3 VPN configuration, perform one of the following tasks:

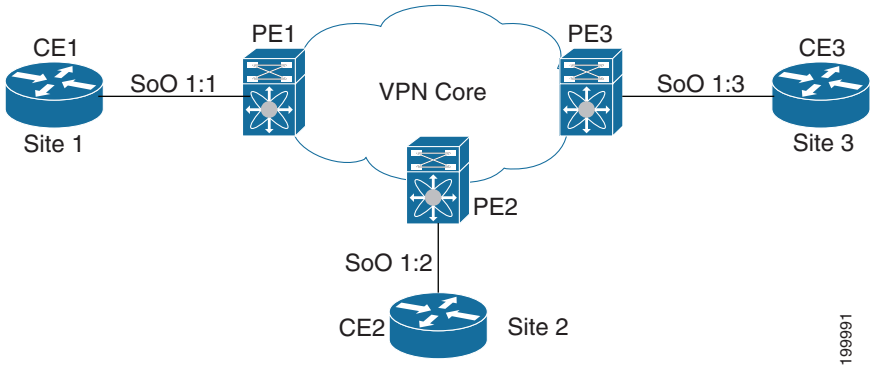
Command	Purpose
ping {**host**-*name* | *system-address*} [**vrf** *vrf-name*]	Verifies the connectivity from one CE router to another.
show bgp {**vpnv4** | **vpnv6**} **unicast** [*ip-prefix/length* [**community** *community*] [**community-list** *community-list*] [**dampening**] [**extcommunity** *extcommunity*] [**extcommunity-lis**t *extcommunity-list*] [**filter-list** *filter-list*] [**flap-statistics**] [**neighbors** *neighbor*] [**nexthop** [*nexthop*]] [**regexp** *regexp*] [**imported**] [**exported**] [**summary**] [**labels**]] {**vrf** {*vrf-name* | **all**} | **rd** *route-distinguisher*}	Displays VPN routes from the BGP table.
show bgp ipv6 unicast [**vrf** *vrf-name*]	Displays information about BGP on a VRF for 6VPE.
show forwarding {**ip** | **ipv6**} **route vrf** *vrf-name*	Displays the IP forwarding table that is associated with a VRF. Check that the loopback addresses of the local and remote CE routers are in the routing table of the PE routers.
show {**ip** | **ipv6**} **bgp** [**vrf** *vrf-name*]	Displays information about BGP on a VRF.
show ip ospf *instance-tag* **vrf** *vrf-name*	Displays information about the Routing Information Protocol (RIP).
show ip ospf sham-links vrf *vrf-name*	Displays the operational status of all sham links that are configured for the device.
show ip route [*ip-address* [*mask*]] [*protocol*] **vrf** *vrf-name*	Displays the current state of the routing table. Use the *ip-address* argument to verify that CE1 has a route to CE2. Verify the routes learned by CE1. Make sure that the route for CE2 is listed.
show {**ip** | **ipv6**} **route vrf** *vrf-name*	Displays the IP routing table that is associated with a VRF. Check that the loopback addresses of the local and remote CE routers are in the routing table of the PE routers.
show running-config bgp	Displays the running configuration for BGP.
show running-config vrf ***vrf-name***	Displays the running configuration for VRFs.
show vrf *vrf-name* **interface** *if-type*	Verifies the route distinguisher (RD) and interface that are configured for the VRF.
trace *destination* [**vrf** *vrf-name*]	Discovers the routes that packets take when traveling to their destination. The **trace** command can help isolate a trouble spot if two routers cannot communicate.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Configuration Examples for MPLS Layer 3 VPNs

This section uses the following sample MPLS network shown in Figure 20-3.

Figure 20-3 Sample MPLS Layer3 Network



- Example: MPLS Layer 3 VPN Using BGP, page 20-373
- , page 20-374
- Example: MPLS Layer 3 VPN Using Static or Direct Routes, page 20-376
- Example: MPLS Layer 3 VPN Using OSPF, page 20-378
- Example: MPLS Layer 3 VPN Using EIGRP, page 20-378
- Example: MPLS 6VPE Using BGP, page 20-379
- Example: Hub-and-Spoke Topology, page 20-380
- Example: OSPF Sham-Link Support for an MPLS VPN, page 20-382
- Example: Enabling MPLS on the specified interface, page 20-383

Note All examples show the basic configuration required for the PE router and the CE router.

Example: MPLS Layer 3 VPN Using BGP

The following example shows how to configure an MPLS Layer 3 VPN using BGP:

PE Configuration	CE Configuration
vrf context vpn1 rd 100:1 address-family ipv4 unicast route-target export 100:1 route-target import 100:1 ! interface Loopback0 ip address 61.61.0.61/32 ! interface Ethernet 2/1 vrf member vpn1 ip address 31.0.0.61/24 ! feature bgp feature-set mpls feature mpls l3vpn router bgp 100] router-id 61.61.0.61 log-neighbor-changes neighbor 62.62.0.62 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended neighbor 63.63.0.63 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended ! vrf vpn1 neighbor 31.0.0.31 remote-as 150 address-family ipv4 unicast as-override soo 1:1 ! ! Note: as-override at PE or allowas-in at CE is required if all CEs use the same remote AS number. ! !if all CE sites are using the same BGP AS number, one of the following scheme must be used: !- configure BGP as-override at the PE !- configure disable-peer-as-check at the PE and allowas-in at the CE	! interface Ethernet2/1 ip address 31.0.0.31/24 ! feature bgp router bgp 150 log-neighbor-changes neighbor 31.0.0.61 remote-as 100 address-family ipv4 unicast !

Example: MPLS Layer 3 VPN Using RIP

The following example shows how to configure an MPLS Layer 3 VPN using RIP:

PE Configuration	CE Configuration
vrf context vpn1 rd 100:1 address-family ipv4 unicast route-target export 100:1 route-target import 100:1 ! interface Loopback0 ip address 61.61.0.61/32 ! feature rip router rip Test1 vrf vpn1 address-family ipv4 unicast redistribute bgp 100 route-map rmap1 ! interface Ethernet2/1 vrf member vpn1 site-of-origin 1:1 ip address 31.0.0.61/24 ip router rip Test1 ! feature bgp feature-set mpls feature mpls l3vpn router bgp 100 router-id 61.61.0.61 log-neighbor-changes neighbor 62.62.0.62 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended neighbor 63.63.0.63 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended ! vrf vpn1 address-family ipv4 unicast redistribute rip Test1 route-map rmap1 ! route-map rmap1 permit 10 !	! feature rip router rip Test1 ! interface Ethernet 2/1 ip address 31.0.0.31/24 ip router rip Test1 !

Example: MPLS Layer 3 VPN Using Static or Direct Routes

The following example shows how to configure an MPLS Layer 3 VPN using static or direct routes:

PE Configuration	CE Configuration
PE1 route-map allow permit 10 vrf context vpn1 rd 100:1 address-family ipv4 unicast route-target import 100:1 route-target export 100:1 router bgp 100 neighbor 100.1.1.2 remote-as 100 address-family vpnv4 unicast send-community extended update-source loopback0 vrf vpn1 address-family ipv4 unicast redistribute direct route-map allow redistribute static route-map allow ! static route to CE vrf context vpn1 ip route 11.10.10.0/24 11.0.0.2 ! ! PE-CE link interface Ethernet2/1 vrf member vpn1 ip address 11.0.0.1/24 no shutdown ! ! Loopback for iBGP vpnv4 neighborship interface loopback0 ip address 100.1.1.1/32 ip router ospf 1 area 0.0.0.0 !PE2 route-map allow permit 10 vrf context vpn1 rd 100:1 address-family ipv4 unicast route-target import 100:1 route-target export 100:1 router bgp 100 neighbor 100.1.1.1 remote-as 100 address-family vpnv4 unicast send-community extended update-source loopback0 vrf vpn1 address-family ipv4 unicast redistribute direct route-map allow redistribute static route-map allow ! ! static route to CE vrf context vpn1 ip route 12.10.10.0/24 12.0.0.2 ! ! PE-CE link interface Ethernet2/1 vrf member vpn1 ip address 12.0.0.1/24 no shutdown ! ! Loopback for iBGP vpnv4 neighborship interface loopback0 ip address 100.1.1.2/32 ip router ospf 1 area 0.0.0.0	CE1 ! ! Static default route to PE ! ip route 0.0.0.0/0 11.0.0.1 ! ! PE-CE link ! interface Ethernet2/1 ip address 11.0.0.2/24 no shutdown ! ! Loopback on CE to test static link between PE-CE ! interface loopback11 ip address 11.10.10.1/24 CE2 ! ! Static default route to PE ! ip route 0.0.0.0/0 12.0.0.1 ! ! PE-CE link ! interface Ethernet2/1 ip address 12.0.0.2/24 no shutdown ! ! Loopback on CE to test static link between PE-CE ! interface loopback12 ip address 12.10.10.1/24

Example: MPLS Layer 3 VPN Using OSPF

The following example shows how to configure a MPLS Layer 3 VPN using OSPF:

PE Configuration	CE Configuration
vrf context vpn1 rd 100:1 address-family ipv4 unicast route-target export 100:1 route-target import 100:1 ! feature ospf router ospf o1 vrf vpn1 address-family ipv4 unicast redistribute bgp 100 route-map rmap1 ! interface Loopback0 ip address 61.61.0.61/32 ! interface Ethernet 2/1 vrf member vpn1 ip address 31.0.0.61/24 ip router ospf o1 area 0.0.0.0 ! feature bgp feature-set mpls feature mpls l3vpn router bgp 100 router-id 61.61.0.61 log-neighbor-changes neighbor 62.62.0.62 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended neighbor 63.63.0.63 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended ! vrf vpn1 address-family ipv4 unicast redistribute ospf o1 route-map rmap1 ! route-map rmap1 permit 10 !	! feature ospf router ospf o1 ! interface Ethernet 2/1 ip address 31.0.0.31/24 ip router ospf o1 area 0.0.0.0 !

Example: MPLS Layer 3 VPN Using EIGRP

The following example shows how to configure a MPLS Layer3 VPN using OSPF:

PE Configuration	CE Configuration
vrf context vpn1 rd 100:1 address-family ipv4 unicast route-target export 100:1 route-target import 100:1 ! feature eigrp router eigrp 200 vrf vpn1 address-family ipv4 unicast redistribute bgp 100 route-map rmap1 ! interface Loopback0 ip address 61.61.0.61/32 ! interface Ethernet 2/1 vrf member vpn1 site-of-origin 1:1 ip address 31.0.0.61/24 ip router eigrp 200 ! feature bgp feature-set mpls feature mpls l3vpn router bgp 100 router-id 61.61.0.61 log-neighbor-changes neighbor 62.62.0.62 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended neighbor 63.63.0.63 remote-as 100 update-source Loopback0 address-family vpnv4 unicast send-community extended ! vrf vpn1 address-family ipv4 unicast redistribute eigrp 200 route-map rmap1 ! route-map rmap1 permit 10 !	! feature eigrp router eigrp 200 ! interface Ethernet 2/1 ip address 31.0.0.31/24 ip router eigrp 200 !

Example: MPLS 6VPE Using BGP

The following example shows how to configure MPLS 6VPE using BGP:

PE Configuration	CE Configuration
vrf context vpn1 rd 100:1 address-family ipv6 unicast route-target export 100:1 route-target import 100:1 ! interface Loopback0 ip address 61.61.0.61/32 ! interface Ethernet 2/1 vrf member vpn1 ip address 68:9::61/64 ! feature bgp feature-set mpls feature mpls l3vpn router bgp 100 router-id 61.61.0.61 log-neighbor-changes neighbor 62.62.0.62 remote-as 100 update-source Loopback0 address-family vpnv6 unicast send-community extended neighbor 63.63.0.63 remote-as 100 update-source Loopback0 address-family vpnv6 unicast send-community extended ! vrf vpn1 neighbor 68:9::31 remote-as 150 address-family ipv6 unicast as-override soo 1:1 ! route-map map1 permit10 ! Note: as-override at PE or allowas-in at CE is required if all CEs use the same remote AS number. ! !if all CE sites are using the same BGP AS number, one of the following scheme must be used: !- configure BGP as-override at the PE !- configure disable-connected-check at the PE and allowas-in at the CE	! interface Ethernet 2/1 ipv6 address 68:9::31/64 ! feature bgp router bgp 150 log-neighbor-changes neighbor 68:9::61 remote-as 100 address-family ipv6 unicast !

Example: Hub-and-Spoke Topology

The following example shows how to configure a hub-and-spoke configuration for an IPv4 MPLS network with eBGP configured between the hub PE3 and hub CE3 routers. It uses the sample hub-and-spoke topology shown in Figure 20-2.

configuration at hub PE3:

```
!Import/export
vrf context 2hub
 rd 1:103
 address-family ipv4 unicast
  route-target export 1:103
  route-target import 1:103
  route-target import 1:101
  route-target import 1:102
!
vrf context 2spokes
 address-family ipv4 unicast
  route-target export 1:100
!BGP
feature bgp
feature mpls l3vpn
router bgp 100
 log-neighbor-changes
 neighbor 62.62.0.62 remote-as 100
  update-source Loopback0
  address-family vpnv4 unicast
   send-community extended
 neighbor 61.61.0.61 remote-as 100
  update-source Loopback0
  address-family vpnv4 unicast
   send-community extended
!
vrf 2hub
  neighbor 33.0.0.33 remote-as 150
  address-family ipv4 unicast
!
 vrf 2spokes
  neighbor 33.0.1.33 remote-as 150
  address-family ipv4 unicast
   allowas-in 3
```

configuration at hub CE3:

```
feature bgp
router bgp 150
 log-neighbor-changes
!2hub
 neighbor 33.0.0.63 remote-as 100
  address-family ipv4 unicast

!2spokes
 neighbor 33.0.1.63 remote-as 100
  address-family ipv4 unicast
   disable-peer-as-check
```

configuration at spoke PE1:

```
!Import/export
vrf context spoke
 rd 1:101
 address-family ipv4 unicast
  route-target export 1:101
  route-target import 1:101
  route-target import 1:100
```

configuration at spoke PE2:

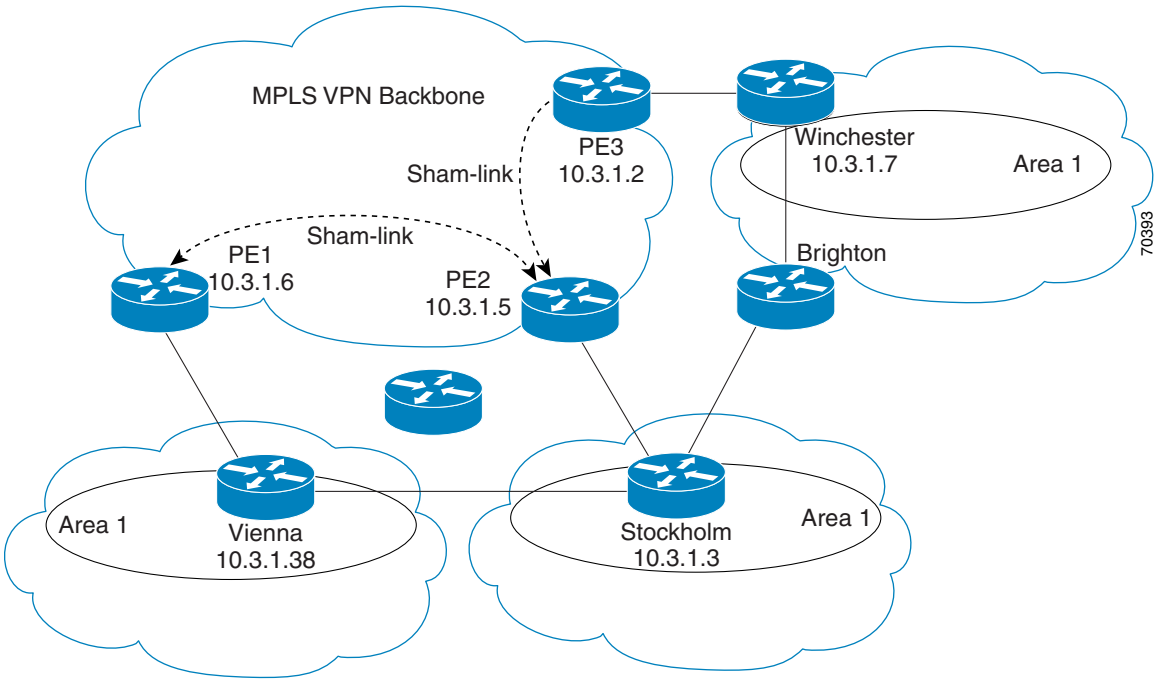
```
!Import/export

vrf context spoke
 rd 1:102
```

```
 address-family ipv4 unicast
  route-target export 1:102
  route-target import 1:102
  route-target import 1:100
!
router bgp 100
 log-neighbor-changes
 neighbor 63.63.0.63 remote-as 100
  update-source Loopback0
  address-family vpnv4 unicast
   send-community extended
!if all CE sites are using the same BGP AS number,
! you must perform the following tasks:
!- configure BGP as-override and allowas-in at the PE
```
!- configure disable-peer-as-check at the CE

Example: OSPF Sham-Link Support for an MPLS VPN

The example in this section shows how to use a sham link to affect only the OSPF intra-area path selection of the PE and CE devices. The PE device also uses the information received from the multiprotocol Border Gateway Protocol (MP-BGP) to set the outgoing label stack of incoming packets and to decide the egress PE device to which the packets must be label switched.

The figure below shows a sample MPLS VPN topology in which a sham-link configuration is necessary. A VPN client has three sites, each with a backdoor link. Two sham links have been configured, one between PE-1 and PE-2, and another between PE-2 and PE-3. A sham link between PE-1 and PE-3 is not necessary in this configuration because the Vienna and Winchester sites do not share a backdoor link.



The following example shows how to configure sham links and a demand circuit:

```
switch(config)# feature-set mpls
switch(config)# feature mpls l3vpn
switch(config)# feature ospf
```

```
switch(config)# device ospf class
switch(config-device)# vrf class1
switch(config-device-vrf)# redistribute bgp 100 route-map allow
switch(config-device-vrf)# area 11 sham-link 10.0.0.1 10.0.0.2
switch(config-device-vrf-slink)# demand-circuit
switch(config-device-vrf-slink) # end
```

The following example show how to display the configuration values for demand circuit in sham links for VRF value class1:

```
switch# sh ip ospf sham-links vrf class1
    SL1-0.0.0.0-10.0.0.1-10.0.0.2 line protocol is up
        IP address 10.0.0.1, Process ID 100 VRF class1, area 0.0.0.0
        State P2P, Network type P2P, cost 1
        Run as demand circuit
        Index 3, Transmit delay 1 sec
        0 Neighbors, flooding to 0, adjacent with 0
        Timer intervals: Hello 10, Dead 40, Wait 40, Retransmit 5
        No authentication
        Number of opaque link LSAs: 0, checksum sum 0
        Adjacency Information :
        Destination IP address: 10.0.0.2
```

The following example show how to display the configuration values for a demand circuit in sham links for all VRFs:

```
switch# show ip ospf sham-links vrf all
    SL1-0.0.0.0-10.0.0.1-10.0.0.2 line protocol is up
        IP address 10.0.0.1, Process ID class VRF class1, area 0.0.0.11
        State P2P, Network type P2P, cost 1
        Run as demand circuit
        Index 1, Transmit delay 1 sec
        0 Neighbors, flooding to 0, adjacent with 0
        Timer intervals: Hello 10, Dead 40, Wait 40, Retransmit 5
        No authentication
        Number of opaque link LSAs: 0, checksum sum 0
        Adjacency Information :
        Destination IP address: 10.0.0.2
    SL2-0.0.0.0-10.0.0.1-10.0.0.2 line protocol is up
        IP address 10.0.0.1, Process ID class VRF blue, area 0.0.0.11
        State P2P, Network type P2P, cost 1
        Run as demand circuit
        Index 2, Transmit delay 1 sec
        0 Neighbors, flooding to 0, adjacent with 0
        Timer intervals: Hello 10, Dead 40, Wait 40, Retransmit 5
        No authentication
        Number of opaque link LSAs: 0, checksum sum 0
        Adjacency Information :
        Destination IP address: 10.0.0.
```

Example: Enabling MPLS on the specified interface

The following example show how to enable MPLS on the interface:

```
switch(config)# int e1/1
switch(config-if)# no switchport
switch(config-if)# mpls ip forwarding

switch(config-subif)# int vlan 5
switch(config-if)# mpls ip forwarding

switch(config-if)# int po5.1
```

```
switch(config-subif)# mpls ip forwarding
ERROR: MPLS is not supported on subinterfaces. Instead enable MPLS on the parent
interface.

switch(config)# int po5
switch(config-if)# mpls ip forwarding
```

Additional References for MPLS Layer 3 VPNs

For additional information related to implementing MPLS Layer 3 VPNs, see the following sections:

- Related Documents, page 20-384
- MIBs, page 20-384

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
VRF-aware services	*Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide*

MIBs

MIBs	MIBs Link
MPLS-L3VPN-STD-MIB	To locate and download Cisco MIBs, go to the following URL: https://cfnng.cisco.com/mibs

Feature History for MPLS Layer 3 VPNs

Table 20-2 lists the release history for this feature.

Table 20-2 Feature History for MPLS Layer 3 VPNs

Feature Name	Releases	Feature Information
MPLS Layer 3 VPNs	7.3(0)DX(1)	Added support in M3 Series modules.
OSPF Sham-Link Support for MPLS VPN	6.2(2)	This feature allows you to use a sham link to connect VPN client sites that run Open Shortest Path First (OSPF) and share back door OSPF links in a Multiprotocol Label Switching (MPLS) VPN configuration. The following commands were introduced or modified: **demand-circuit**.
MPLS Layer 3 VPNs	5.2(7)	Added matching and setting support for import maps on standard and extended communities for Cisco NX-OS Release 5.2(7) and later 5.2 releases.

Table 20-2 Feature History for MPLS Layer 3 VPNs (continued)

Feature Name	Releases	Feature Information
MPLS Layer 3 VPNs	5.2(5)	Removed the MPLS license requirement for the EIGRP site of origin feature.
MPLS Layer 3 VPNs	5.2(1)	This feature was introduced.
6VPE	5.2(1)	This feature was introduced.

Configuring MPLS Layer 3 VPN Label Allocation

This chapter describes how to configure label allocation for Multiprotocol Label Switching (MPLS) Layer 3 virtual private networks (L3VPNs) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS L3VPN Label Allocation

The MPLS provider edge (PE) router stores both local and remote routes and includes a label entry for each route. By default, Cisco NX-OS uses per-prefix label allocation which means that each prefix is assigned a label. For distributed platforms, the per-prefix labels consume memory. When there are many VPN routing and forwarding instances (VRFs) and routes, the amount of memory that the per-prefix labels consume can become an issue.

You can enable per-VRF label allocation to advertise a single VPN label for local routes throughout the entire VRF. The router uses a new VPN label for the VRF decoding and IP-based lookup to learn where to forward packets for the PE or customer edge (CE) interfaces.

You can enable different label allocation modes for Border Gateway Protocol (BGP) Layer 3 VPN routes to meet different requirements and to achieve trade-offs between scalability and performance. All labels are allocated within the global label space. Cisco NX-OS supports the following label allocation modes:

- Per-prefix—A label is allocated for each VPN prefix. VPN packets received from remote PEs can be directly forwarded to the connected CE that advertised the prefix, based on the label forwarding table. However, this mode also uses many labels. This mode is the only mode available when VPN packets sent from PE to CE are label switched. This is the default label allocation mode.
- Per-VRF—A single label is assigned to all local VPN routes in a VRF. This mode requires an IPv4 or IPv6 lookup in the VRF forwarding table once the VPN label is removed at the egress PE. This mode is the most efficient in terms of label space as well as BGP advertisements, and the lookup does not result in any performance degradation. Cisco NX-OS uses the same per-VRF label for both IPv4 and IPv6 prefixes.

Note EIBGP load balancing is not supported for a VRF that uses per-VRF label mode.

- Aggregate Labels—BGP can allocate and advertise a local label for an aggregate prefix. Forwarding requires an IPv4 or IPv6 lookup that is similar to the per-VRF scenario. A single per-VRF label is allocated and used for all prefixes that need a lookup.
- VRF connected routes—When directly connected routes are redistributed and exported, an aggregate label is allocated for each route. The packets that come in from the core are decapsulated and a lookup is done in the VRF IPv4 or IPv6 table to determine whether the packet is for the local router or for another router or host that is directly connected. A single per-VRF label is allocated for all such routes.
- Label hold down—When a local label is no longer associated with a prefix, to allow time for updates to be sent to other PEs, the local label is not released immediately. A ten minute hold down timer is started per label. Within this hold down period, the label can be reclaimed for the prefix. When the timer expires, BGP releases the label.

Per-VRF Label Allocation Mode

The following conditions apply when you configure per-VRF label allocation:

- The VRF uses one label for all local routes.
- When you enable per-VRF label allocation, any existing per-VRF aggregate label is used. If no per-VRF aggregate label is present, the software creates a new per-VRF label.

  The CE does not lose data when you disable per-VRF label allocation because the configuration reverts to the default per-prefix labeling configuration.
- A per-VRF label forwarding entry is deleted only if the VRF, BGP, or address family configuration is removed.

IPv6 Label Allocation

IPv6 prefixes are advertised with the allocated label to iBGP peers that have the labeled-unicast address-family enabled. The received eBGP next hop is not propagated to such peers; instead, the local IPv4 session address is sent as an IPv4-mapped IPv6 next hop. The remote peer resolves this next hop through one or more IPv4 MPLS LSPs in the core network.

You can use a route reflector to advertise the labeled 6PE prefixes between PEs. You must enable the labeled-unicast address-family between the route reflector and all such peers. The route reflector does not need to be in the forwarding path and propagates the received next hop as is to iBGP peers and route reflector clients.

Note 6PE also supports both per-prefix and per-VRF label allocation modes, as in 6VPE.

Licensing Requirements for MPLS L3VPN Label Allocation

Product	License Requirement
Cisco NX-OS	L3VPN label allocation requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS L3VPN Label Allocation

L3VPN label allocation has the following prerequisites:

- Ensure that you have configured MPLS, and LDP or RSVP TE in your network. All routers in the core, including the PE routers, must be able to support MPLS forwarding.
- Ensure that you have installed the correct license for MPLS and any other features you will be using with MPLS.
- Ensure that you disable the external/internal Border Gateway Protocol (BGP) multipath feature if it is enabled before you configure per-VRF label allocation mode.
- Before configuring a 6VPE per VRF label, ensure that the IPv6 address family is configured on that VRF.

Guidelines and Limitations for MPLS L3VPN Label Allocation

L3VPN label allocation has the following configuration guidelines and limitations:

- F Series modules do not natively support label switching. They can leverage M Series modules for label switching using proxy forwarding. For more information on proxy forwarding, see the *Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide*.
- Enabling per-VRF label allocation causes BGP reconvergence, which can result in data loss for traffic coming from the MPLS VPN core.

Note You can minimize network disruption by enabling per-VRF label allocation during a scheduled MPLS maintenance window. Also, if possible, avoid enabling this feature on a live router.

- Per-prefix MPLS counters for VPN prefixes are lost when you enable per-VRF label allocation.
- Aggregate labels and per-VRF labels are global across all virtual device contexts (VDCs) and are in a separate, dedicated label range.
- Aggregate prefixes for per-prefix label allocation share the same label in a given VRF.

Default Settings for MPLS L3VPN Label Allocation

Table 21-1 lists the default settings for L3VPN label allocation parameters.

Table 21-1 Default L3VPN Label Allocation Parameters

Parameters	Default
L3VPN feature	Disabled
Label allocation mode	Per prefix

Configuring MPLS L3VPN Label Allocation

This section includes the following topics:

Configuring Per-VRF L3VPN Label Allocation Mode

You can configure per-VRF L3VPN label allocation mode for Layer 3 VPNs.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature mpls l3vpn**
5. **router bgp** *as-number*

6. **vrf** *vrf-name*
7. **address-family** {**ipv6** | **ipv4**}{**unicast** | **multicast**}
8. **label-allocation-mode per-vrf**
9. (Optional) **show bgp l3vpn detail vrf** *vrf-name*
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 5	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1` `switch(config-router-vrf)#`	Enters router VRF configuration mode. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**address-family** {**ipv4** | **ipv6**} **unicast** | **multicast**} **Example:** `switch(config-router-vrf)# address-family ipv6 unicast` `switch(config-router-vrf-af)#`	Specifies the IP address family type and enters address family configuration mode.
Step 8	**label-allocation-mode per-vrf** **Example:** `switch(config-router-vrf-af)# label-allocation-mode per-vrf`	Allocates labels on a per-VRF basis.

	Command	Purpose
Step 9	**show bgp l3vpn detail vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show bgp l3vpn detail vrf vpn1`	(Optional) Displays information about Layer 3 VPN configuration on BGP for this VRF. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 10	**copy running-config startup-config** **Example:** `switch(config-router-vrf-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Allocating Labels for IPv6 Prefixes in the Default VRF

If you are running IPv6 over an IPv4 MPLS core network (6PE), you can allocate labels for the IPv6 prefixes in the default VRF.

Note By default, labels are not allocated for IPv6 prefixes in the default VRF.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature mpls l3vpn**
5. **router bgp** *as-number*
6. **address-family ipv6** {**unicast** | **multicast**}
7. **allocate-label** {**all** | **route-map** *route-map*}
8. (Optional) **show running-config bgp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.

	Command	Purpose
Step 3	**feature-set mpls** **Example:** switch(config)# feature-set mpls	Enables the MPLS feature-set.
Step 4	**feature mpls l3vpn** **Example:** switch(config)# feature mpls l3vpn	Enables the MPLS Layer 3 VPN feature.
Step 5	**router bgp** *as-number* **Example:** switch(config)# router bgp 1.1 switch(config-router)#	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**address-family ipv6** {**unicast** | **multicast**} **Example:** switch(config-router)# address-family ipv6 unicast switch(config-router-af)#	Specifies the IPv6 address family type and enters address family configuration mode.
Step 7	**allocate-label** {**all** | **route-map** *route-map*} **Example:** switch(config-router-af)# allocate-label all	Allocates labels for IPv6 prefixes in the default VRF. • The **all** keyword allocates labels for all IPv6 prefixes. • The **route-map** keyword allocates labels for IPv6 prefixes matched in the specified route map. The route-map can be any case-sensitive alphanumeric string up to 63 characters.
Step 8	**show running-config bgp** **Example:** switch(config-router-af)# show running-config bgp	(Optional) Displays information about the BGP configuration.
Step 9	**copy running-config startup-config** **Example:** switch(config-router-af)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Enabling Sending MPLS Labels in IPv6 over an IPv4 MPLS Core Network (6PE) for iBGP Neighbors

You can enable sending MPLS labels to iBGP neighbors.

Note The **address-family ipv6 labeled-unicast** command is supported only for iBGP neighbors. You cannot use this command with the **address-family ipv6 unicast** command.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature mpls l3vpn**
5. **router bgp** *as-number*
6. **neighbor** *ip-address*
7. **address-family ipv6 labeled-unicast**
8. (Optional) **show running-config bgp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 5	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 209.165.201.1` `switch(config-router-neighbor)#`	Adds an entry to the BGP or multiprotocol BGP neighbor table. The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation.

	Command	Purpose
Step 7	**address-family ipv6 labeled-unicast** **Example:** switch(config-router-neighbor)# address-family ipv6 labeled-unicast switch(config-router-neighbor-af)#	Specifies IPv6 labeled unicast address prefixes. This command is accepted only for iBGP neighbors.
Step 8	**show running-config bgp** **Example:** switch(config-router-neighbor-af)# show running-config bgp	(Optional) Displays information about the BGP configuration.
Step 9	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Verifying MPLS L3VPN Label Allocation Configuration

To display the L3VPN label allocation configuration, perform one of the following tasks:

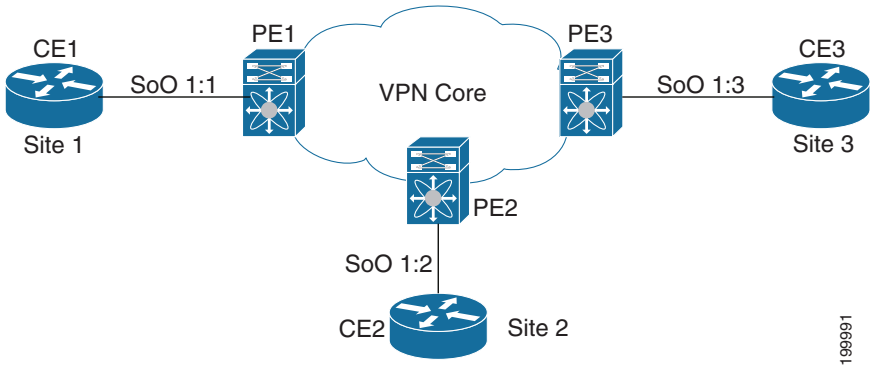
Command	Purpose
show bgp l3vpn [**detail**] [**vrf** *vrf-name*]	Displays Layer 3 VPN information for BGP in a VRF.
show bgp vpnv4 unicast labels [**vrf** *vrf-name*]	Displays label information for BGP.
show ip route [**vrf** *vrf-name*]	Displays label information for routes.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Configuration Examples for MPLS L3VPN Label Allocation

This section uses the following sample MPLS network shown in Figure 21-1.

Figure 21-1 Sample MPLS Layer3 Network



The following example shows how to configure per-VRF label allocation for an IPv4 MPLS network.

PE Configuration

```
PE1
-----
  vrf context vpn1
    rd 100:1
    address-family ipv4 unicast
      route-target export 200:1
  router bgp 100
   neighbor 10.1.1.2 remote-as 100
      address-family vpnv4 unicast
         send-community extended
      update-source loopback10
   vrf vpn1
      address-family ipv4 unicast
         label-allocation-mode per-vrf
      neighbor 36.0.0.2 remote-as 300
         address-family ipv4 unicast
```

Additional References for MPLS L3VPN Label Allocation

For additional information related to implementing L3VPN Label Allocation, see the following sections:

- Related Documents, page 21-396
- MIBs, page 21-396

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIBs	MIBs Link
MPLS-L3VPN-STD-MIB	To locate and download Cisco MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Feature History for MPLS L3VPN Label Allocation

Table 21-2 lists the release history for this feature.

Table 21-2 Feature History for L3VPN Label Allocation

Feature Name	Releases	Feature Information
Per-VRF label allocation	5.2(1)	This feature was introduced.

Configuring MPLS Layer 3 VPN Load Balancing

This chapter describes how to configure load balancing for Multiprotocol Label Switching (MPLS) Layer 3 virtual private networks (VPNs) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS Layer 3 VPN Load Balancing

Load balancing distributes traffic so that no individual router is overburdened. In an MPLS Layer 3 network, you can achieve load balancing by using the Border Gateway Protocol (BGP). When multiple iBGP paths are installed in a routing table, a route reflector advertises only one path (next hop). If a router is behind a route reflector, all routes that are connected to multihomed sites are not advertised

unless a different route distinguisher is configured for each virtual routing and forwarding instance (VRF). (A route reflector passes learned routes to neighbors so that all iBGP peers do not need to be fully meshed.)

This section includes the following topics:

iBGP Load Balancing

When a BGP-speaking router configured with no local policy receives multiple network layer reachability information (NLRI) from the internal BGP (iBGP) for the same destination, the router chooses one iBGP path as the best path and installs the best path in its IP routing table. iBGP load balancing enables the BGP-speaking router to select multiple iBGP paths as the best paths to a destination and to install multiple best paths in its IP routing table.

eBGP Load Balancing

When a router learns two identical eBGP paths for a prefix from a neighboring autonomous system, it chooses the path with the lower route ID as the best path. The router installs this best path in the IP routing table. You can enable eBGP load balancing to install multiple paths in the IP routing table when the eBGP paths are learned from a neighboring autonomous system instead of picking one best path.

During packet switching, depending on the switching mode, the router performs either per-packet or per-destination load balancing among the multiple paths.

Layer 3 VPN Load Balancing

Layer 3 VPN load balancing for both eBGP and iBGP allows you to configure multihomed autonomous systems and provider edge (PE) routers to distribute traffic across both external BGP (eBGP) and iBGP multipaths.

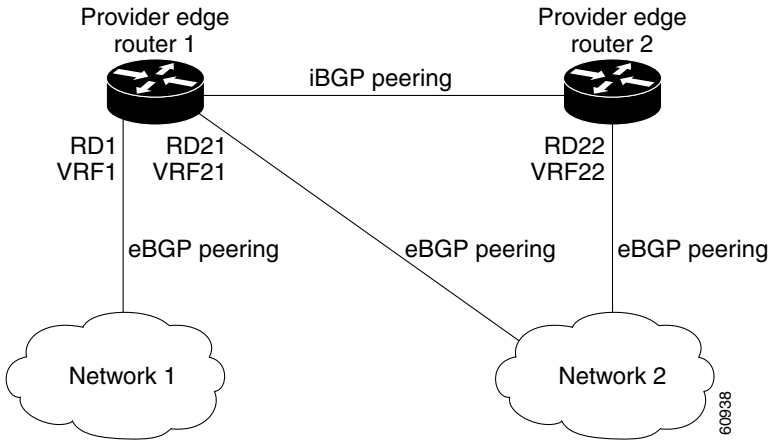
Layer 3 VPN load balancing supports IPv4 and IPv6 for the PE routers and VPNs.

BGP installs up to the maximum number of multipaths allowed. BGP uses the best path algorithm to select one path as the best path, inserts the best path into the routing information base (RIB) and advertises the best path to BGP peers. The router can insert other paths into the RIB but selects only one path as the best path.

Layer 3 VPNs load balance on a per-packet or per-source or destination pair basis. To enable load balancing, configure the router with Layer 3 VPNs that contain VPN routing and forwarding instances (VRFs) that import both eBGP and iBGP paths. You can configure the number of paths separately for each VRF.

Figure 22-1 shows an MPLS provider network that uses BGP. In the figure, two remote networks are connected to PE1 and PE2, which are both configured for VPN unicast iBGP peering. Network 2 is a multihomed network that is connected to PE1 and PE2. Network 2 also has extranet VPN services configured with Network 1. Both Network 1 and Network 2 are configured for eBGP peering with the PE routers.

Figure 22-1 Provider MPLS Network Using BGP



You can configure PE1 so that it can select both iBGP and eBGP paths as multipaths and import these paths into the VPN routing and forwarding instance (VRF) of Network 1 to perform load balancing.

Traffic is distributed as follows:

- IP traffic that is sent from Network 2 to PE1 and PE2 is sent across the eBGP paths as IP traffic.
- IP traffic that is sent from PE1 to PE2 is sent across the iBGP path as MPLS traffic.
- Traffic that is sent across an eBGP path is sent as IP traffic.

Any prefix that is advertised from Network 2 will be received by PE1 through route distinguisher (RD) 21 and RD22.

- The advertisement through RD21 is carried in IP packets.
- The advertisement through RD22 is carried in MPLS packets.

The router can select both paths as multipaths for VRF1 and insert these paths into the VRF1 RIB.

Layer 3 VPN Load Balancing with Route Reflectors

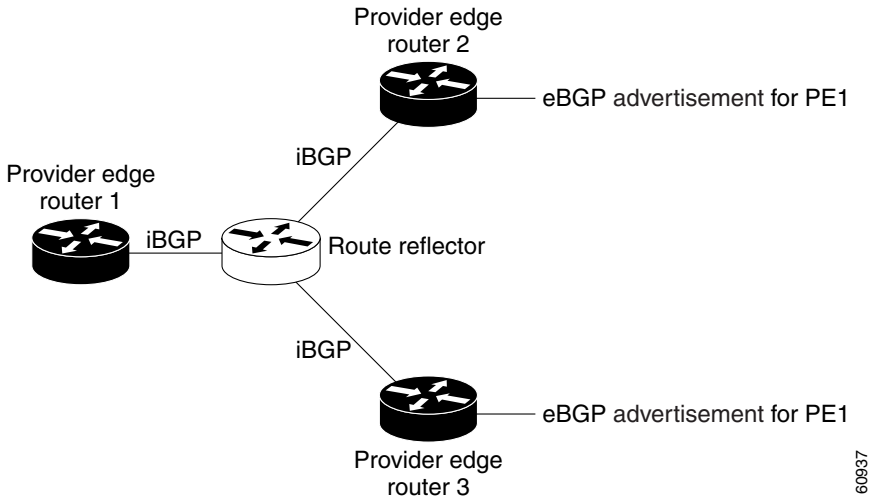
Route reflectors reduce the number of sessions on PE routers and increase the scalability of Layer 3 VPN networks. Route reflectors hold on to all received VPN routes to peer with PE routers. Different PEs can require different route target-tagged VPNv4 and VPNv6 routes. The route reflector may also need to send a refresh for a specific route target to a PE when the VRF configuration has changed. Storing all routes increases the scalability requirements on a route reflector. You can configure a route reflector to only hold routes that have a defined set of route target communities.

You can configure route reflectors to service a different set of VPNs and configure a PE to peer with all route reflectors that service the VRFs configured on the PE. When you configure a new VRF with a route target that the PE does not already hold routes for, the PE issues route refreshes to the route reflectors and retrieves the relevant VPN routes.

Note The route reflectors do not need to be in the forwarding path, but you must configure unique route distinguisher (RDs) for VPN sites that are multihomed.

Figure 22-2 shows a topology that contains three PE routers and a route reflector, all configured for iBGP peering. PE2 and PE3 each advertise an equal preference eBGP path to PE1. By default, the route reflector chooses only one path and advertises PE1.

Figure 22-2 Topology with a Route Reflector



For all equal preference paths to PE1 to be advertised through the route reflector, you must configure each VRF with a different RD. The prefixes received by the route reflector are recognized differently and advertised to PE1.

Layer 2 Load Balancing Coexistence

The load balance method that is required in the Layer 2 VPN is different from the methos that is used for Layer 3 VPN. Layer 3 VPN and Layer 2 VPN forwarding is performed independently using two different types of adjacencies. The forwarding is not impacted by using a different method of load balancing for the Layer 2 VPN.

Note Load balancing is not supported at the ingress PE for Layer 2 VPNs.

BGP VPNv4 Multipath

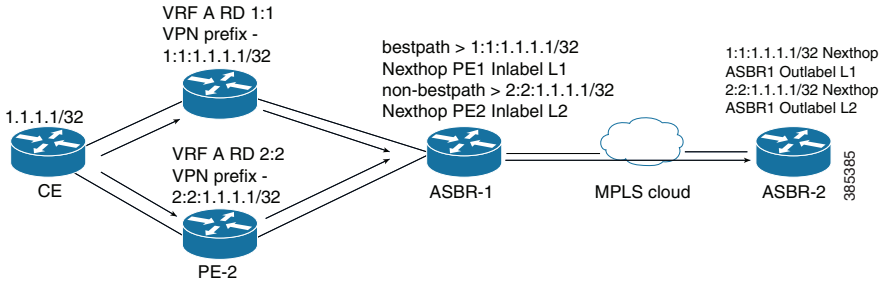
Starting from Cisco NX-OS Release 8.0(1), use the BGP VPNv4 Multipath feature to achieve Equal Cost Multi-Path (ECMP) for traffic flowing from an Autonomous System Border Router (ASBR) towards the Provider Edge (PE) device in an Multi-Protocol Label Switching (MPLS) cloud network by using a lower number of prefixes and MPLS labels. This feature configures the maximum number of multipaths for both eBGP and iBGP paths. This feature can be configured on PE devices and Route Reflectors in an MPLS topology.

Consider a scenario in which a dual homed Customer Edge (CE) device is connected to 2 PE devices and you have to utilize both the PE devices for traffic flow from ASBR-2 to the CE device.

Currently, as shown in Figure 22-3, Virtual Routing and Forwarding (VRF) on each PE is configured using separate Route Distinguishers (RD). The CE device generates a BGP IPv4 prefix. The PE devices are configured with 2 separate RDs and generate two different VPN-IPv4 prefixes for the BGP IPv4 prefix sent by the CE device. ASBR-1 receives both the VPN-IPv4 prefixes and adds them to the routing

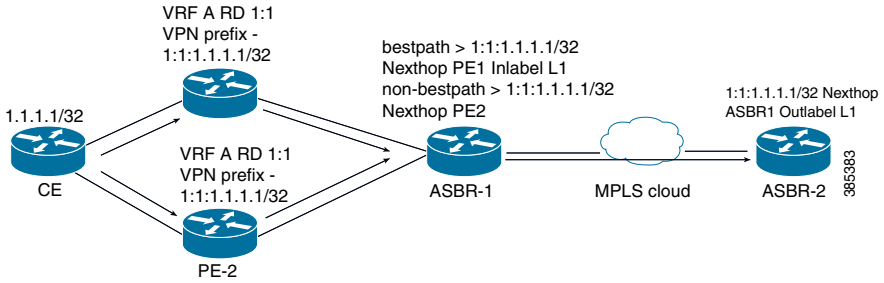
table. ASBR-1 allocates Inter-AS option-B labels, Inlabel L1 and Inlabel L2, to both the VPN routes and then advertises both VPN routes to ASBR-2. To use both PE devices to maintain traffic flow, ASBR-1 has to utilize two Inter-AS option-B labels and two prefixes which limits the scale that can be supported.

Figure 22-3 Virtual Routing and Forwarding (VRF) on each PE configured using separate Route Distinguishers



Using the BGP VPN Multipath feature, as shown in Figure 22-4, you can enable the VRF on both PE devices to use the same RD. In such a scenario, ASBR-1 receives the same prefix from both the PE devices. ASBR-1 allocates only one Inter-AS option-B label, Inlabel L1, to the received prefix and advertises the VPN route to ASBR-2. In this case, the scale is enhanced as traffic flow using both PE devices is established with only one prefix and label on ASBR-1.

Figure 22-4 Enabling the VRF on both PE devices to use the same RD



BGP Cost Community

The BGP cost community is a nontransitive extended community attribute that is passed to iBGP and confederation peers but not to eBGP peers. (A confederation is a group of iBGP peers that use the same autonomous system number to communicate to external networks.) The BGP cost community attributes includes a cost community ID and a cost value. You can customize the BGP best path selection process for a local autonomous system or confederation by configuring the BGP cost community attribute. You configure the cost community attribute in a route map with a community ID and cost value. BGP prefers the path with the lowest community ID, or for identical community IDs, BGP prefers the path with the lowest cost value in the BGP cost community attribute.

BGP uses the best path selection process to determine which path is the best where multiple paths to the same destination are available. You can assign a preference to a specific path when multiple equal cost paths are available.

Since the administrative distance of iBGP is worse than the distance of most Interior Gateway Protocols (IGPs), the unicast Routing Information Base (RIB) may apply the same BGP cost community compare algorithm before using the normal distance or metric comparisons of the protocol or route. VPN routes that are learned through iBGP can be preferred over locally learned IGP routes.

The cost extended community attribute is propagated to iBGP peers when an extended community exchange is enabled.

How the BGP Cost Community Influences the Best Path Selection Process

The cost community attribute influences the BGP best path selection process at the point of insertion (POI). The POI follows the IGP metric comparison. When BGP receives multiple paths to the same destination, it uses the best path selection process to determine which path is the best path. BGP automatically makes the decision and installs the best path into the routing table. The POI allows you to assign a preference to a specific path when multiple equal cost paths are available. If the POI is not valid for local best path selection, the cost community attribute is silently ignored.

You can configure multiple paths with the cost community attribute for the same POI. The path with the lowest cost community ID is considered first. All of the cost community paths for a specific POI are considered, starting with the one with the lowest cost community ID. Paths that do not contain the cost community (for the POI and community ID being evaluated) are assigned with the default community cost value.

Applying the cost community attribute at the POI allows you to assign a value to a path originated or learned by a peer in any part of the local autonomous system or confederation. The router can use the cost community as a tie breaker during the best path selection process. You can configure multiple instances of the cost community for separate equal cost paths within the same autonomous system or confederation. For example, you can apply a lower cost community value to a specific exit path in a network with multiple equal cost exits points, and the BGP best path selection process prefers that specific exit path.

Cost Community and EIGRP PE-CE with Back-Door Links

BGP prefers back-door links in an Enhanced Interior Gateway Protocol (EIGRP) Layer 3 VPN topology if the back-door link is learned first. A back-door link, or a route, is a connection that is configured outside of the Layer 3 VPN between a remote and main site.

The pre-best path point of insertion (POI) in the BGP cost community supports mixed EIGRP Layer 3 VPN network topologies that contain VPN and back-door links. This POI is applied automatically to EIGRP routes that are redistributed into BGP. The pre-best path POI carries the EIGRP route type and metric. This POI influences the best-path calculation process by influencing BGP to consider this POI before any other comparison step.

Note You can configure BGP path selection to ignore the cost community but this is not recommended for configuration with a back-door link.

Licensing Requirements for MPLS Layer 3 VPN Load Balancing

Product	License Requirement
Cisco NX-OS	MPLS Layer 3 VPN load balancing requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.*

Prerequisites for MPLS Layer 3 VPN Load Balancing

MPLS Layer 3 VPN load balancing has the following prerequisites:

- You must enable the MPLS and L3VPN features.
- You must install the correct license for MPLS.

Guidelines and Limitations for MPLS Layer 3 VPN Load Balancing

MPLS Layer 3 VPN load balancing has the following configuration guidelines and limitations:

- If you place a router behind a route reflector and it is connected to multihomed sites, the router will not be advertised unless separate VRFs with different RDs are configured for each VRF.
- Each IP routing table entry for a BGP prefix that has multiple iBGP paths uses additional memory. We recommend that you do not use this feature on a router with a low amount of available memory or when it is carrying a full Internet routing table.
- You should not ignore the BGP cost community when a back-door link is present and EIGRP is the PE-CE routing protocol.
- Starting with Cisco NX-OS Release 8.0(1), a maximum of 500K VPN prefixes is supported on M3-Series I/O modules. For M2-Series I/O modules, a maximum of 125K VPN prefixes is supported.
- Starting with Cisco NX-OS Release 8.0(1), 4K VRFs are supported.

Default Settings for MPLS Layer 3 VPN Load Balancing

Table 22-1 lists the default settings for MPLS Layer 3 VPN load balancing parameters.

Table 22-1 Default MPLS Layer 3 VPN Load Balancing Parameters

Parameters	Default
Layer 3 VPN feature	Disabled
BGP cost community ID	128
BGP cost community cost	2147483647
maximum multipaths	1
BGP VPNv4 Multipath	Disabled

Configuring MPLS Layer 3 VPN Load Balancing

This section includes the following topics:

- Configuring BGP Load Balancing for eBGP and iBGP, page 22-405
- Configuring BGPv4 Multipath, page 22-407

Configuring BGP Load Balancing for eBGP and iBGP

You can configure a Layer 3 VPN load balancing for an eBGP or iBGP network.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature bgp**
5. **router bgp** *as-number*
6. (Optional) **bestpath cost-community ignore**
7. **address-family** {**ipv4** | **ipv6**} **unicast**
8. **maximum-paths [ibgp]** *number-of-paths*
9. (Optional) **show running-config bgp**
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the Layer 3 VPN feature.
Step 4	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 5	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**bestpath cost-community ignore** **Example:** `switch(config-router)# bestpath` `cost-community ignore#`	(Optional) Ignores the cost community for BGP bestpath calculations.
Step 7	**address-family** {**ipv4** | **ipv4**} **unicast** **Example:** `switch(config-router)# address-family ipv4` `unicast` `switch(config-router-af)#`	Enters address family configuration mode for configuring IP routing sessions.
Step 8	**maximum-paths** [**ibgp**] *number-of-paths* **Example:** `switch(config-router-af)# maximum-paths 4`	Configures the maximum number of multipaths allowed. Use the **ibgp** keyword to configure iBGP load balancing. The range is from 1 to 16.
Step 9	**show running-config bgp** **Example:** `switch(config-router-af)# show` `running-config bgp`	(Optional) Displays the running configuration for BGP.
Step 10	**copy running-config startup-config** **Example:** `switch(config-router-af)# copy` `running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring BGPv4 Multipath

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **router bgp** *as-number*
4. **address-family vpnv4 unicast**
5. **maximum-paths eibgp** *parallel-paths*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**router bgp** *as-number* **Example:** `switch(config)# router bgp 2` `switch(config-router)#`	Assigns an autonomous system (AS) number to a router and enter the router BGP configuration mode.
Step 4	**address-family vpnv4 unicast** **Example:** `switch(config-router)# address-family vpnv4 unicast` `switch(config-router-af)#`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard VPNv4 address prefixes:
Step 5	**maximum-paths eibgp** *parallel-paths* **Example:** `switch(config-router-af)# maximum-paths eibgp 3`	Specifies the maximum number of BGP VPNv4 multipaths for both eBGP and iBGP paths. The range is from 1 to 32.

Configuring BGP Cost Community

You can configure the BGP cost community for routes in a Layer 3 VPN.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**

2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature bgp**
5. **route-map** ***map-name*** {**permit | deny**} [*sequence-number*]
6. **set extcommunity cost** [**igp | pre-bestpath**] *community-id cost-value*
7. **router bgp** *as-number*
8. (Optional) **router-id** *ip-address*
9. **neighbor** *ip-address* **remote-as** *as-number*
10. **address-family** {**vpnv4 | vpnv6**} **unicast**
11. **send-community extended**
12. **route-map** ***map-name*** {**in | out**}
13. (Optional) **show running-config bgp**
14. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the Layer 3 VPN feature.
Step 4	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 5	**route-map** *map-name* {**permit** | **deny**} [*sequence-number*] **Example:** `switch(config)# route-map costMap permit 10`	Enters route map configuration mode to create or configure a route map.

	Command	Purpose
Step 6	**set extcommunity cost** [**igp** | **pre-bestpath**] *community-id cost-value* **Example:** `switch(config-route-map)# set extcommunity cost 1 100`	Creates a set clause to apply the cost community attribute. Multiple cost community set clauses can be configured in each route map block or sequence. Each cost community set clause must have a *community-id* (0 to 255). The cost community set clause with the lowest *cost-value* is preferred by the best path selection process when all other attributes are equal. The *cost-value* range is from 0 to 4294967295. Paths that are not configured with the cost community attribute are assigned the default *cost-value* of 2147483647.
Step 7	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 8	**router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.255`	(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 9	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 192.0.2.1 remote-as 100` `switch(config-router-neighbor)#`	Configures a neighbor and enters address family neighbor configuration mode. The *ip-address* is an IPv4 or IPv6 address. The *as-number* can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 10	**address-family** {**vpnv4** | **vpnv6**} **unicast** **Example:** `switch(config-router-neighbor)#` `address-family vpnv4 unicast` `switch(config-router-neighbor-af)#`	Enters address family configuration mode for configuring IP VPN sessions.
Step 11	**send-community extended** **Example:** `switch(config-router-neighbor-af)#` `send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor.
Step 12	**route-map** *map-name* {**in** | **out**} **Example:** `switch(config-router-neighbor-af)#` `route-map costMap in`	Configures a route map for this address family.

	Command	Purpose
Step 13	**show running-config bgp** **Example:** `switch(config-router-neighbor-af)# show running-config bgp`	(Optional) Displays the running configuration for BGP.
Step 14	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Verifying the MPLS Layer 3 VPN Load-Balancing Configuration

To display an MPLS Layer 3 VPN load balancing configuration, perform the following task:

Command	Purpose
show running-config bgp	Displays the running configuration for BGP.
show bgp vpnv4 unicast	Display the VPNv4 routes from the BGP table.

For detailed information about the fields in the output from these commands, see the *Cisco NX-OS MPLS Command Reference*.

Configuration Examples for MPLS Layer 3 VPN Load Balancing

This section includes the following topics:

Example: MPLS Layer 3 VPN Load Balancing

The following example shows how to configure iBGP load balancing:

```
configure terminal
 feature-set mpls
 feature mpls l3vpn
 feature bgp
 router bgp 1.1
  bestpath cost-community ignore
  address-family ipv6 unicast
   maximum-paths ibgp 4
```

Example: BGP VPNv4 Multipath

The following example shows how to configure a maximum of 3 BGP VPNv4 multipaths:

```
configure terminal
 router bgp 100
 address-family vpnv4 unicast
```

```
  maximum-paths eibgp 3
```

Example: MPLS Layer 3 VPN Cost Community

The following example shows how to configure the BGP cost community:

```
configure terminal
 feature-set mpls
 feature mpls l3vpn
 feature bgp
 route-map CostMap permit
  set extcommunity cost 1 100
 router bgp 1.1
  router-id 192.0.2.255
  neighbor 192.0.2.1 remote-as 1.1
   address-family vpnv4 unicast
    send-community extended
    route-map CostMap in
```

Additional References for MPLS Layer 3 VPN Load Balancing

For additional information related to implementing an Layer 3 VPN load balancing, see the following sections:

- Related Documents, page 22-412
- MIBs, page 22-412

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIBs	MIBs Link
MPLS-Layer 3 VPN-STD-MIB	To locate and download Cisco MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Feature History for MPLS Layer 3 VPN Load Balancing

Table 22-2 lists the release history for this feature.

Table 22-2 Feature History for MPLS Layer 3 VPN Load Balancing

Feature Name	Releases	Feature Information
BGP VPNv4 Multipath	8.0(1)	This feature was introduced.
Layer 2 and Layer 3 load balancing co-existance	6.2(2)	This feature was introduced.
Layer 3 VPN load balancing	5.2(1)	This feature was introduced.
BGP cost community	5.2(1)	This feature was introduced.

Configuring MPLS over GRE

This chapter describes how to configure a Virtual Private Network (VPN) generic routing encapsulation (GRE) tunnel for moving Multiprotocol Label Switching (MPLS) packets over a non-MPLS network.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About Configuring MPLS over GRE

This section includes the following topics:

PE-to-PE GRE Tunneling

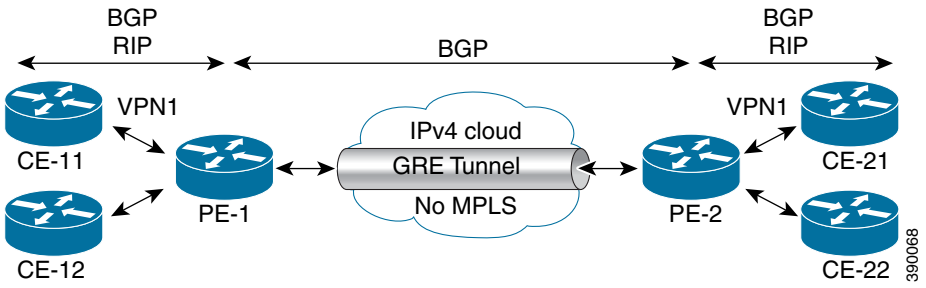
A provider-edge-to-provider-edge (PE-to-PE) tunnel provides a scalable way to connect multiple customer networks across a non-MPLS network. With this configuration, traffic that is destined to multiple customer networks is multiplexed through a single generic routing encapsulation (GRE) tunnel. A similar nonscalable alternative is to connect each customer network through separate GRE tunnels (for example, connecting one customer network to each GRE tunnel).

The PE devices assign virtual routing and forwarding (VRF) numbers to the customer edge (CE) devices on each side of the non-MPLS network. The PE devices use routing protocols such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), or Routing Information Protocol (RIP) to learn about the IP networks behind the CE devices. The routes to the IP networks behind the CE devices are stored in the VRF routing table of the associated CE device.

The PE device on one side of the non-MPLS network uses routing protocols (that operate within the non-MPLS network) to learn about the PE device on the other side of the non-MPLS network. The learned routes that are established between the PE devices are then stored in the main or default routing table. PE device on the other side of the network uses BGP to learn about the routes that are associated with the customer networks that are associated with the PE devices. These learned routes are not known to the non-MPLS network.

The following figure shows BGP defining a route to the BGP neighbor (the opposing PE device) through the GRE tunnel that spans the non-MPLS network. Because routes that are learned by the BGP neighbor include the GRE tunnel next hop, all customer network traffic is sent using the GRE tunnel.

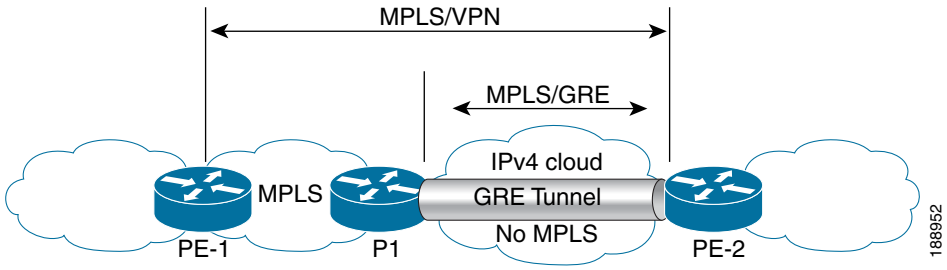
Figure 23-1 PE-to-PE GRE Tunnel



P-to-PE Tunneling

As shown in the figure below, the provider-to-provider-edge (P-to-PE) tunneling configuration provides a way to connect a PE device (P1) to a Multiprotocol Label Switching (MPLS) segment (PE-2) across a non-MPLS network. In this configuration, MPLS traffic that is destined to the other side of the non-MPLS network is sent through a single generic routing encapsulation (GRE) tunnel.

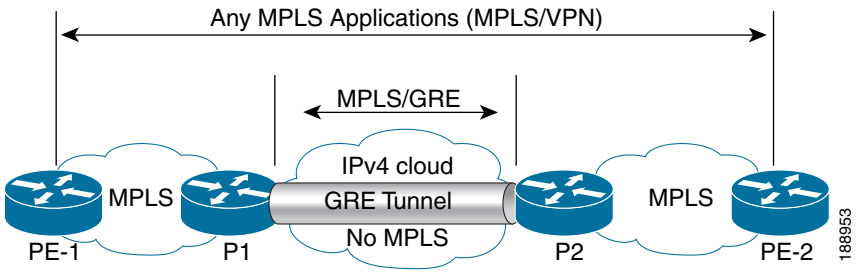
Figure 23-2 P-to-PE GRE Tunnel



P-to-P Tunneling

As shown in the figure below, the provider-to-provider (P-to-P) configuration provides a method of connecting two Multiprotocol Label Switching (MPLS) segments (P1 to P2) across a non-MPLS network. In this configuration, MPLS traffic that is destined to the other side of the non-MPLS network is sent through a single generic routing encapsulation (GRE) tunnel.

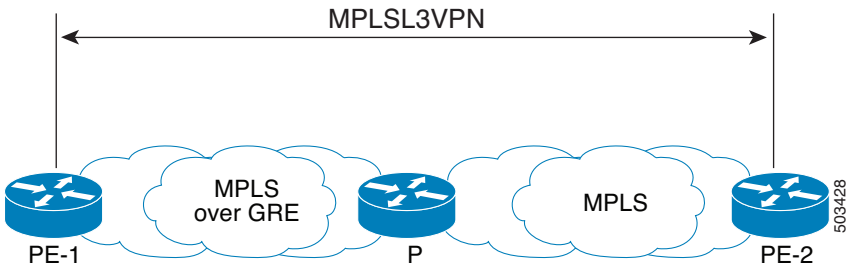
Figure 23-3 P-to-P Tunnel



MPLS over GRE Tunnel with MPLS Stitching

The figure given below shows the scenario of MPLS over GRE Tunnel with MPLS Stitching.

Figure 23-4 MPLS over GRE + MPLS Stiching



- The GRE Tunnel is between PE-1 and P. MPLS is configured on the tunnels, thereby enabling MPLS over GRE tunnel.
- When traffic is from PE-1 to PE-2, the tunnel encapsulation happens at PE-1 and the decapsulation happens at P. The tunnel terminates on P.
- Normal MPLS is configured between P and PE-2.

The tunnel can be configured in following scenarios:

- Tunnel is between PE to PE.
- Tunnel is between PE to P.
- Tunnel is between P to P.

Licensing Requirements for MPLS on GRE

The following table shows the licensing requirements for this feature:

Product	License Requirement
NX-OS	MPLS Layer 3 and Layer 2 VPNs require an MPLS license. For a complete explanation of the NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for Configuring MPLS over GRE

- Ensure that your MPLS VPN is configured and working properly.

Guidelines and Limitations for Configuring MPLS over GRE

- Starting from Cisco NX-OS Release 8.3(1), MPLS over GRE is supported on M3-Series I/O modules.
- MPLS over GRE is supported on M1-Series and M2-Series I/O modules.
- Layer 2 VPN over GRE is not supported on M3 series modules.
- MPLS over GRE is not supported on F3 series modules.
- The Tunnel stitching with MPLS GRE feature is supported only on M3 modules from Cisco NX-OS Release 8.4(2). This feature requires reload if you upgrade from releases prior to Cisco NX-OS Release 8.4(2).

Layer 3 VPN MPLS over GRE does not support the following:

- MPLS over GRE Inter AS Option C and the generic Option C are not supported.
- Quality of service (QoS) service policies that are configured on the tunnel interface. QoS service policies are supported on the physical interface or subinterface.
- GRE options—Sequencing, checksum, and source route.
- IPv6 generic routing encapsulation (GRE).
- Advance features such as Carrier Supporting Carrier (CSC) and Interautonomous System (Inter-AS).

- GRE-based Layer 3 VPN does not interwork with MPLS or IP VPNs. From Cisco NX-OS Release 8.4(2), the interwork support is applicable to M3 series modules.
- GRE tunnel is supported only as a core link (PE-PE, PE-P, P-P, P-PE). A Provide-Edge to Customer-Edge (PE-CE) link is not supported.
- IPv6 VPN forwarding using GRE tunnels.
- Static route mapping to GRE tunnels.
- Bidirectional Forwarding Detection (BFD) with GRE tunnels.

Layer 2 VPLS over GRE has the following configuration guidelines and limitations:

- A VPLS instance must be configured on each Provider Edge (PE) device.
- Load balancing at the Virtual Private LAN Service (VPLS) ingress or at the core is not supported for flood or multicast traffic.
- Virtual circuit connection verification (VCCV) over flow aware transport of MPLS pseudowires (FAT PW) is not supported. The Interior Gateway Protocol (IGP) load balancing for VCCV is also unsupported.

Ethernet over MPLS over GRE has the following configuration guidelines and limitations:

- Multiple point-to-point tunnels can saturate the physical link with routing information if bandwidth is not configured correctly on a tunnel interface.
- A tunnel may have a recursive routing problem if routing is not configured accurately. The best path to a tunnel destination through the tunnel itself; therefore recursive routing causes the tunnel interface to flap. To avoid recursive routing problems, keep control-plane routing separate from tunnel routing by using the following methods:
  - Use a different autonomous system number or tag.
  - Use a different routing protocol.
  - Ensure that static routes are used to override the first hop (watch for routing loops).
- The following error is displayed when there is recursive routing to a tunnel destination:

```
%TUN-RECURDOWN Interface Tunnel 0 temporarily disabled due to recursive routing
```

Configuring MPLS over GRE

This section includes the following topics:

Configuring Layer 3 VPN Configuring MPLS over GRE

To configure a generic routing encapsulation (GRE) tunnel and create a virtual point-to-point link across the non-MPLS network, you must perform this task on the devices located at both ends of the GRE tunnel.

SUMMARY STEPS

1. **feature mpls**

2. **feature tunnel**
3. **configure terminal**
4. **interface tunnel** *tunnel-number*
5. **ip address** *ip-address ip-address-mask*
6. **mpls ip**
7. **tunnel source** *source-address*
8. **tunnel destination** *destination-address*
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**feature mpls** **Example** `switch# feature mpls`	Enables MPLS-related features.
Step 2	**feature tunnel** **Example:** `switch# feature tunnel`	Enables tunnel-related features.
Step 3	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 4	**interface tunnel** *tunnel-number* **Example:** `switch(config)# interface tunnel 1` `switch(config-if)#`	Enters interface configuration mode and creates a tunnel interface. The range for the *tunnel-number* argument is from 0 to 4095.
Step 5	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 10.0.0.1 255.255.255.0 3`	Assigns an IP address to this tunnel interface.
Step 6	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 10.0.0.1 255.255.255.0 3`	Assigns an IP address to this tunnel interface.
Step 7	**tunnel source** *source-address* **Example:** `switch(config-if)# tunnel source 10.1.1.1`	Specifies the IP address of the tunnel source.

	Command	Purpose
Step 8	**tunnel destination** *destination-address* **Example:** `switch(config-if)# tunnel source 10.1.1.2`	Specifies the IP address of the tunnel destination. • For provider edge (PE)-to-PE tunneling, configure tunnels with the same destination address.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-vrf-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring Layer 2 VPN Configuring MPLS over GRE

Restrictions

You cannot have two tunnels using the same encapsulation mode with exactly the same source and destination addresses. The work around is to create a loopback interface and source packets from the loopback interface. This restriction is applicable only for generic routing encapsulation (GRE) tunnels.

SUMMARY STEPS

1. **configure terminal**
2. **interface loopback** *number*
3. **ip address** *ip-address mask*
4. **exit**
5. **interface tunnel** *number*
6. **tunnel mode gre**
7. **interface tunnel** *number*
8. **ip address** *ip-address mask*
9. **tunnel source** {*ip-address* | *type/number*}
10. **tunnel destination** {*hostname* | *ip-address*}
11. **mpls ip** {**propagate-ttl** | **ttl-expiration pop** [*labels*]}
12. **exit**
13. **ip route** *prefix mask interface-type interface-number*
14. **ip route** *prefix mask interface-type interface-number*
15. [**no**] **l2vpn vfi context** *context-name*
16. (Optional) **description** *description*
17. **vpn** *vpn-id*
18. **member peer** ip-address [*vc-id*] **encapsulation mpls**
19. **vlan configuration** *vlan-id*
20. **member vfi** *context-name*
21. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface loopback** *number* **Example:** `switch(config)# interface Loopback 0` `switch(config-if)#`	Enters interface configuration mode and configures a loopback interface. • The range of the *number* argument is from 0 to 1023.
Step 3	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 209.165.202.157 255.255.255.224`	Configures a primary address for this interface.
Step 4	**exit** **Example:** `switch(config-if)# exit` `switch(config)#`	Exits interface configuration mode and returns to global configuration mode.
Step 5	**interface tunnel** *number* **Example:** `switch(config)# interface tunnel 1` `switch(config-if)#`	Enters interface configuration mode and configures a tunnel interface. • The range of the *number* argument is from 0 to 4095. • A tunnel should be independent of the endpoint physical interface type, such as TM, Gigabit, Packet over SONET (POS), and TenGigabit. • Up to 100 GRE tunnels are supported.
Step 6	**tunnel mode gre** **Example:** `switch(config-if)# tunnel mode gre`	Sets the encapsulation mode for the tunnel interface.
Step 7	**interface tunnel** *number* **Example:** `switch(config-if)# interface Tunnel 0`	Enters interface configuration mode and configures a tunnel interface. • The range of the *number* argument is from 0 to 4095.

	Command or Action	Purpose
Step 8	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 209.165.200.225 255.255.255.224`	Configures a primary address for this interface.
Step 9	**tunnel source** {*ip-address* | *type*/*number*} **Example:** `switch(config-if)# tunnel source 192.0.0.2`	Sets the source address for a tunnel interface. • The source address is either an explicitly defined IP address or the IP address assigned to the specified interface. • GRE tunnel encapsulation and decapsulation for multicast packets are handled by the hardware. Each hardware-assisted tunnel must have a unique source. Hardware-assisted tunnels cannot share a source even if the destinations are different. You should use secondary addresses on loopback interfaces or create multiple loopback interfaces to ensure that the hardware-assisted tunnels do not share a source.
Step 10	**tunnel destination** {*hostname* | *ip-address*] **Example:** `switch(config-if)# tunnel destination 192.0.0.3`	Specifies the destination for the tunnel interface.
Step 11	**mpls ip** {**propagate-ttl** | **ttl-expiration-pop** [*labels*]} **Example:** `switch(config-if)# mpls ip propagate-ttl`	Controls the generation of the time-to-live (TTL) field in the Multiprotocol Label Switching (MPLS) header. • The **propagate-ttl** keyword enables MPLS forwarding along normally routed paths for the interface. • The **ttl-expiration-pop** keyword specifies that packets with an expired time-to-live (TTL) value are forwarded through the original label stack. • The *labels* argument is the maximum number of labels in the packet necessary for the packet to be forwarded by means of the global IP routing table.
Step 12	**exit** **Example:** `switch(config-if)# exit switch(config)#`	Exits interface configuration mode and returns to global configuration mode.
Step 13	**ip route** *prefix mask interface-type interface-number* **Example:** `switch(config)# ip route 209.165.201.6 255.255.255.224 tunnel 1`	Creates a static route for routing packets for the designated network to the specified interface.

	Command or Action	Purpose
Step 14	**ip route** *prefix mask interface-type interface-number* **Example:** `switch(config)# ip route 209.165.201.7 255.255.255.224 tunnel 2`	Creates a static route for routing packets for the designated network to the specified interface.
Step 15	[**no**] **l2vpn vfi context** *context-name* **Example:** `switch(config)# l2vpn vfi context example` `switch(config-l2vpn-vfi)#`	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) context between two or more separate networks. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 16	**description** *description* **Example:** `switch(config-l2vpn-vfi)# description example`	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric characters.
Step 17	**vpn** *vpn-id* **Example:** `switch(config-l2vpn-vfi)# vpn 100`	Configures a VPN ID for a VFI context. • The emulated VCs bound to this Layer 2 VFI use this VPN ID for signaling. • The range of the *vpn-id* argument is from 1 to 4294967295.
Step 18	**member peer** *ip-address* [*vc-id*] **encapsulation mpls** **Example:** `switch(config-l2vpn-vfi)# member peer 192.0.2.8 encapsulation mpls`	Configures a dynamic pseudowire member under the VFI.
Step 19	**vlan configuration** *vlan-id* **Example:** `switch(config-l2vpn-vfi)# vlan configuration 200` `switch(config-vlan-config)#`	Enters VLAN configuration mode and creates a VLAN ID.

	Command or Action	Purpose
Step 20	**member vfi** *context-name* **Example:** `switch(config-vlan-config)# member vfi example`	Adds the specified VFI context as a member of this VLAN. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters.
Step 21	**copy running-config startup-config** **Example:** `switch(config-vlan-config)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying Configuring MPLS over GRE

To verify IP tunnel configuration information, perform one of the following tasks:

Command	Purpose
show interface tunnel *number*	Displays the configuration for the tunnel interface (MTU, protocol, transport, and VRF). Displays input and output packets, bytes, and packet rates.
show interface tunnel *number* **brief**	Displays the operational status, IP address, encapsulation type, and MTU of the tunnel interface.
show interface tunnel *number* **description**	Displays the configured description of the tunnel interface.
show interface tunnel *number* **status**	Displays the operational status of the tunnel interface.
show interface tunnel *number* **status err-disabled**	Displays the error disabled status of the tunnel interface.

Configuration Examples for Configuring MPLS over GRE

This section includes the following topics:

Example: Configuring a GRE Tunnel That Spans a Non-MPLS Network

The following example shows how to configure a generic routing encapsulation (GRE) tunnel configuration that spans a non-MPLS network. This example shows the tunnel configuration on the provider edge (PE) devices (PE1 and PE2) located at both ends of the tunnel:

PE1 configuration

```
switch# configure terminal
switch(config)# interface Tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# mpls ip
switch(config-if)# ip address 10.1.1.1 255.255.255.0
switch(config-if)# tunnel source 10.0.0.1
switch(config-if)# tunnel destination 10.0.0.2
switch(config-if)# end
```

PE2 configuration

```
switch# configure terminal
switch(config)# interface Tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# mpls ip
switch(config-if)# ip address 10.1.1.2 255.255.255.0
switch(config-if)# tunnel source 10.0.0.2
switch(config-if)# tunnel destination 10.0.0.1
switch(config-if)# end
```

Example: MPLS Configuration with PE-to-PE GRE Tunnel

The following example is for a basic PE-to-PE tunneling configuration that uses a generic routing encapsulation (GRE) tunnel to span a non-MPLS network:

PE1 configuration

```
feature-set mpls
feature mpls l3vpn
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature bgp

route-map allow permit 10

interface Tunnel0
/* description GRE tunnel */
  mpls ip
  ip address 10.1.1.1/24
  ip router ospf 100 area 0.0.0.0
  tunnel source Ethernet7/12
  tunnel destination 10.131.31.218
  no shutdown

interface Ethernet7/12
/* description Core facing interface */
  mpls ip
  ip address 10.131.31.205/30
  ip router rip 100
  no shutdown

interface loopback0
/* description Loopback for creating router sessions */
  ip address 10.131.31.1/32
  ipv6 address 1:1::1:1/128
  ip router ospf 100 area 0.0.0.0
```

```
interface loopback1
/*description Loopback for creating alternate router sessions */
  ip address 10.131.31.11/32
  ip router ospf 100 area 0.0.0.0

interface loopback11
/* description Loopback for testing vpn forwarding */
  vrf member vpn1
  ip address 1.1.1.1/24
  ipv6 address 11:11::11:1/120

vrf context vpn1
  rd 100:1
  address-family ipv4 unicast
    route-target import 100:1
    route-target export 100:1
  address-family ipv6 unicast
    route-target import 100:1
    route-target export 100:1

router bgp 100
  address-family ipv6 unicast
    redistribute direct route-map allow
    allocate-label all

  neighbor 10.131.31.2 remote-as 100
/* description VPNv4, VPNv6 */
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended

  neighbor 10.131.31.22 remote-as 100
/* description 6PE */
    update-source loopback1
    address-family ipv6 labeled-unicast
      send-community extended

  vrf vpn1
    address-family ipv4 unicast
      redistribute direct route-map allow
    address-family ipv6 unicast
      redistribute direct route-map allow

router ospf 100
router rip 100
```

PE2 configuration

```
feature-set mpls
feature mpls l3vpn
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature bgp

route-map allow permit 10

interface Tunnel0
/* description GRE tunnel */
  mpls ip
```

```
  ip address 10.1.1.2/24
  ip router ospf 100 area 0.0.0.0
  tunnel source Ethernet7/38
  tunnel destination 10.131.31.205
  no shutdown

interface Ethernet7/38
/* description Core facing interface */
  mpls ip
  ip address 10.131.31.218/30
  ip router rip 100
  no shutdown

interface loopback0
/*  description Loopback for creating router sessions */
  ip address 10.131.31.2/32
  ipv6 address 1:1::1:2/128
  ip router ospf 100 area 0.0.0.0

interface loopback1
/* description Loopback for creating alternate router sessions */
  ip address 10.131.31.22/32
  ip router ospf 100 area 0.0.0.0

interface loopback11
/* description Loopback for testing vpn forwarding */
  vrf member vpn1
  ip address 2.2.1.1/24
  ipv6 address 22:22::22:1/120

vrf context vpn1
  rd 100:1
  address-family ipv4 unicast
    route-target import 100:1
    route-target export 100:1
  address-family ipv6 unicast
    route-target import 100:1
    route-target export 100:1

router bgp 100
  address-family ipv6 unicast
    redistribute direct route-map allow
    allocate-label all

  neighbor 10.131.31.1 remote-as 100
/* description VPNv4, VPNv6 */
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended

  neighbor 10.131.31.11 remote-as 100
/* description 6PE */
    update-source loopback1
    address-family ipv6 labeled-unicast
      send-community extended

  vrf vpn1
    address-family ipv4 unicast
      redistribute direct route-map allow
    address-family ipv6 unicast
      redistribute direct route-map allow
```

```
router ospf 100
router rip 100
```

Example: MPLS Configuration with P-to-PE GRE Tunnel

The following example is for a basic P-to- PE tunneling configuration that uses a generic routing encapsulation (GRE) tunnel to span a non-MPLS network:

P configuration

```
feature-set mpls
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature mpls l3vpn

interface Tunnel0
/* description GRE tunnel */
mpls ip
ip address 10.1.1.1/24
ip router ospf 100 area 0.0.0.0
tunnel source Ethernet7/14
tunnel destination 10.131.31.205

interface Ethernet7/14
  mpls ip
  ip address 10.131.31.206/30
  ip router rip 100
  no shutdown

interface Ethernet7/36
  mpls ip
  ip address 10.131.31.217/30
  ip router rip 100
  no shutdown

interface loopback0
  ip address 10.131.31.10/32
  ip router ospf 100 area 0.0.0.0

router rip 100
router ospf 100
```

PE configuration

```
feature-set mpls
feature mpls l3vpn
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature bgp

route-map allow permit 10

interface Tunnel0
/* description GRE tunnel */
  mpls ip
  ip address 10.1.1.2/24
```

```
  ip router ospf 100 area 0.0.0.0
  tunnel source Ethernet7/12
  tunnel destination 10.131.31.206
  no shutdown

interface Ethernet7/12
/* description Core facing interface */
  mpls ip
  ip address 10.131.31.205/30
  ip router rip 100
  no shutdown

interface loopback0
/* description Loopback for creating router sessions */
  ip address 10.131.31.1/32
  ipv6 address 1:1::1:1/128
  ip router ospf 100 area 0.0.0.0

interface loopback1
/* description Loopback for creating alternate router sessions */
  ip address 10.131.31.11/32
  ip router ospf 100 area 0.0.0.0

interface loopback11
/* description Loopback for testing vpn forwarding */
  vrf member vpn1
  ip address 1.1.1.1/24
  ipv6 address 11:11::11:1/120

vrf context vpn1
  rd 100:1
  address-family ipv4 unicast
    route-target import 100:1
    route-target export 100:1
  address-family ipv6 unicast
    route-target import 100:1
    route-target export 100:1

router bgp 100
  address-family ipv6 unicast
    redistribute direct route-map allow
    allocate-label all

  neighbor 10.131.31.2 remote-as 100
/* description VPNv4, VPNv6 */
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended

  neighbor 10.131.31.22 remote-as 100
/* description 6PE */
    update-source loopback1
    address-family ipv6 labeled-unicast
      send-community extended

  vrf vpn1
    address-family ipv4 unicast
      redistribute direct route-map allow
    address-family ipv6 unicast
      redistribute direct route-map allow

router ospf 100
```

```
router rip 100
```

Additional References for Configuring MPLS over GRE

This section includes the following topics:

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Interface commands	*Cisco Nexus 7000 Series NX-OS Interface Command Reference*

MIBs <Optional: remove if not applicable>

MIBs	MIBs Link
• MPLS-L3VPN-STD-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/dc-os/mibs

Feature History for Layer 3 VPN Configuring MPLS over GRE

Table 23-1 lists the release history for this feature.

Remove the second sentence, above, if it does not apply to the table.

Table 23-1 Feature History for Layer 3 VPN Mpls over GRE

Feature Name	Releases	Feature Information
MPLS over GRE to MPLS Stitching	8.4(2)	Starting from Cisco NX-OS Release 8.4(2), the GRE-based Layer 3 VPN interwork support is applicable to M3 series modules.
MPLS over GRE	8.3(1)	Starting from Cisco NX-OS Release 8.3(1), MPLS over GRE is supported on M3-Series I/O modules.
MPLS over GRE	6.2(2)	The MPLS over GRE feature provides a mechanism for tunneling Multiprotocol Label Switching (MPLS) packets over a non-MPLS network.

InterAS Option B

This document explains the different InterAS option B configuration options. The available options are InterAS option B, InterAS option B (with RFC 3107), and InterAS option B *lite*. The limitations of InterAS option B *lite* are taken care of by the InterAS option B feature in the NX-OS 7.2(0)D1(1) version.

The InterAS option B (with RFC 3107) implementation ensures *complete IGP isolation* between the data centers and WAN. When BGP advertises a particular route to ASBR, it also distributes the label which is mapped to that route.

Note While the InterAS option B lite version is available in the 6.2(2) version, the InterAS option B and InterAS option B (with RFC 3107 implementation) options are available in the 7.2(0)D1(1) version.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About InterAS

An autonomous system (AS) is a single network or group of networks that is controlled by a common system administration group and using a single, clearly defined protocol. In many cases, virtual private networks (VPNs) extend to different ASes in different geographical areas. Some VPNs must extend across multiple service providers; these VPNs are called overlapping VPNs. The connection between ASes must be seamless to the customer, regardless of the complexity or location of the VPNs.

InterAS and ASBRs

Separate ASes from different service providers can communicate by exchanging information in the form of VPN IP addresses. The ASBRs use EBGP to exchange that information. The IBGP distributes the network layer information for IP prefixes throughout each VPN and each AS. The following protocols are used for sharing routing information:

- Within an AS, routing information is shared using IBGP.
- Between ASes, routing information is shared using EBGP. EBGP allows service providers to set up an interdomain routing system that guarantees loop-free exchange of routing information between separate ASes.

The primary function of EBGP is to exchange network reachability information between ASes, including information about the list of AS routes. The ASes use EBGP border edge routers to distribute the routes, which includes label-switching information. Each border edge router rewrites the next-hop and MPLS labels.

InterAS configuration supported in this MPLS VPN can include an interprovider VPN, which is MPLS VPNs that include two or more ASes, connected by separate border edge routers. The ASes exchange routes use EBGP, and no IBGP or routing information is exchanged between the ASes.

Exchanging VPN Routing Information

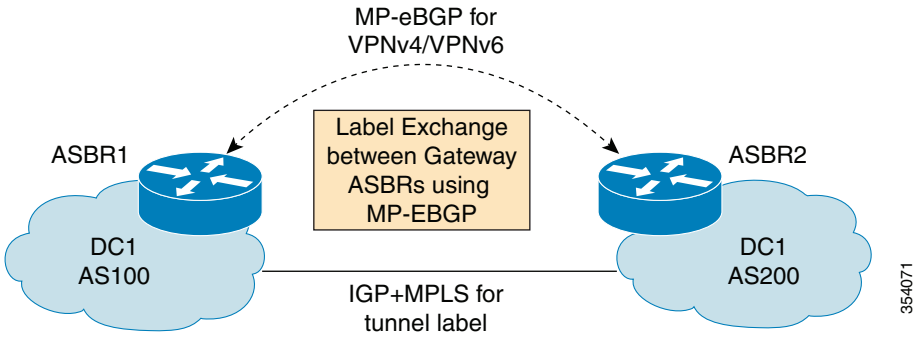
ASes exchange VPN routing information (routes and labels) to establish connections. To control connections between ASes, the PE routers and EBGP border edge routers maintain a label forwarding information base (LFIB). The LFIB manages the labels and routes that the PE routers and EBGP border edge routers receive during the exchange of VPN information.

The ASes use the following guidelines to exchange VPN routing information:

- Routing information includes:
  - The destination network.
  - The next-hop field associated with the distributing router.
  - A local MPLS label.
- A route distinguisher (RD1) is part of a destination network address. It makes the VPN IP route globally unique in the VPN service provider environment.

The ASBRs are configured to change the next-hop when sending VPN NLRIs to the IBGP neighbors. Therefore, the ASBRs must allocate a new label when they forward the NLRI to the IBGP neighbors.

Packet Forwarding



The above figure shows how packets are forwarded between ASes in an interprovider network. A single EBGP connection is established between ASBRs and the ASBRs will exchange routes associated with all local VRFs.

Packets are sent between the ASBRs through MPLS. Packets use the routing information stored in the LFIB of each ASBR.

A data packet carries two levels of labels when it traverses between the ASBR:

- The first label (IGP/Core label) directs the packet to the correct ASBR.
- The second label (VPN route label) directs the packet to the appropriate VRF.

InterAS Options

Nexus 7000/7700 series switches support the following InterAS options:

- **InterAS option A**- In an interAS option A network, autonomous system border router (ASBR) peers are connected by multiple subinterfaces with at least one interface VPN that spans the two ASes. These ASBRs associate each subinterface with a VPN routing and forwarding (VRF) instance and a BGP session to signal unlabeled IP prefixes. As a result, traffic between the back-to-back VRFs is IP. In this scenario, the VPNs are isolated from each other and, because the traffic is IP Quality of Service (QoS) mechanisms that operate on the IP traffic can be maintained. The downside of this configuration is that one BGP session is required for each subinterface (and at least one subinterface is required for each VPN), which causes scalability concerns as the network grows.
- **InterAS option B** - In an interAS option B network, ASBR ports are connected by one or more subinterfaces that are enabled to receive MPLS traffic. A Multiprotocol Border Gateway Router (MP-BGP) session distributes labeled VPN prefixes between the ASBRs. As a result, the traffic that flows between the ASBRs is labeled. The downside of this configuration is that, because the traffic is MPLS, QoS mechanisms that are applied only to IP traffic cannot be carried and the VRFs cannot be isolated. InterAS option B provides better scalability than option A because it requires only one BGP session to exchange all VPN prefixes between the ASBRs. Also, this feature provides nonstop forwarding (NSF) and Graceful Restart. The ASBRs must be directly connected in this option.

Some functions of option B are noted below:

  - You can have an IBGP VPNv4/v6 session between Nexus 7000/7700 series switches within an AS and you can have an EBGP VPNv4/v6 session between data center edge routers and WAN routers.

  - There is no requirement for a per VRF IBGP session between data center edge routers, like in the lite version.
  - LDP distributes IGP labels between ASBRs.
- **InterAS option B (with BGP-3107 or RFC 3107 implementation)**-
  - You can have an IBGP VPNv4/v6 implementation between Nexus 7000/7700 series edge switches within an AS and you can have an EBGP VPNv4/v6 session between data center edge routers and WAN routers.
  - BGP-3107 enables BGP packets to carry label information without using LDP between ASBRs.
  - The label mapping information for a particular route is piggybacked in the same BGP update message that is used to distribute the route itself.
  - When BGP is used to distribute a particular route, it also distributes an MPLS label which is mapped to that route. Many ISPs prefer this method of configuration since it ensures *complete IGP isolation* between the data centers.
- **InterAS option B lite** – Support for the InterAS option B feature is restricted in the Cisco NX-OS 6.2(2) release. Details are noted in the Configuring InterAS Option B (lite version) section.

Licensing Requirements for InterAS Option B

The following table shows the licensing requirements for this feature:

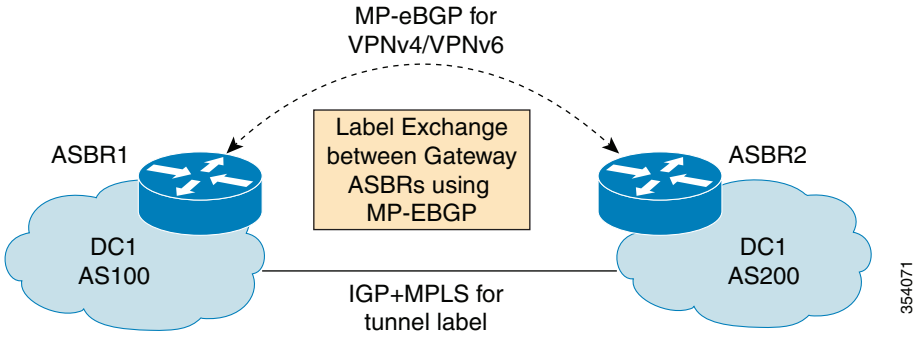
Product	License Requirement
Cisco NX-OS	MPLS Layer 3 requires an MPLS license. For a complete explanation of the NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations for Configuring InterAS Option B

The InterAS option B feature is not supported with BGP confederation AS.

Configuring InterAS Option B

Note The Option B implementation is applicable for the 7.2(0)D1(1) version.



Configuring the Switch for InterAS Option B, page 24-436

Configuring BGP for InterAS Option B, page 24-437

Configuring the Switch for InterAS Option B

You enable certain features on the switch to run InterAS option B.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command). The **install feature-set mpls** command is available only in the default VDC, and you must enable it in default VDC.

Configure VRFs on the DC edge switches with following steps:

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set in the default VDC. **Note** You can only install and enable MPLS in the default VDC. Use the **no** form of this command to uninstall the MPLS feature set.
Step 3	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS LDP feature on the device. **Note** When the MPLS LDP feature is disabled on the device, no LDP commands are available.
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.

	Command	Purpose
Step 5	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 6	**vrf-context** *vrf-name* **Example:** `switch(config)# vrf context VPN1`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**rd** *route-target-ext-community* **Example:** `switch(config-vrf)# rd100:1`	Configures the route distinguisher. The route-distinguisher argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.
Step 9	**route-target {import | export}** *route-target-ext-community* **Example:** `switch(config-vrf-af-ip4)# route-target import 1:1`	Specifies a route-target extended community for a VRF as follows: • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities.
Step 10	**copy running-config startup-config** **Example:** `switch(config-vrf-af-ip4)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

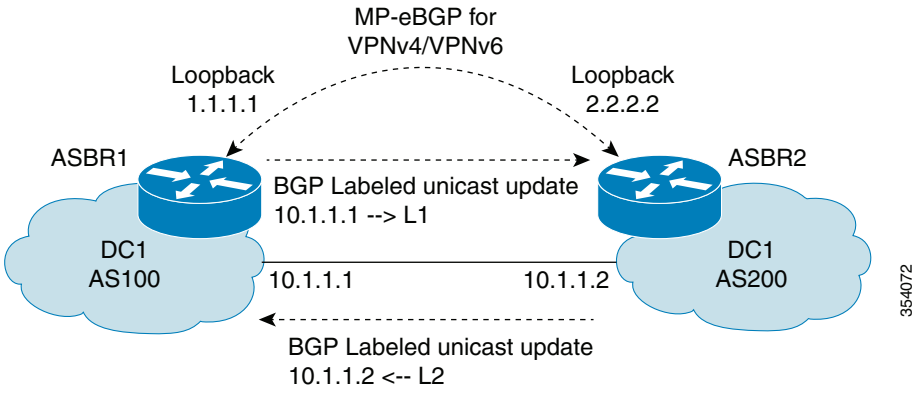
Configuring BGP for InterAS Option B

To configure BGP for InterAS option B, you need to enable this configuration on both the IBGP and EBGP sides. Refer to Figure 1 for reference.

Configure DC Edge switches with IBGP & EBGP VPNv4/v6 with the following steps:

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**router bgp** *as-number* **Example:** `switch(config)# router bgp 100`	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device.
Step 3	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 10.0.0.2`	Adds an entry to the BGP or multiprotocol BGP neighbor table, and enters router BGP neighbor configuration mode.
Step 4	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 200`	The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 5	**address-family {vpnv4 | vpnv6} unicast** **Example:** `switch(config-router-neighbor)# address-family vpnv4 unicast`	Enters address family configuration mode for configuring IP VPN sessions.
Step 6	**send-community {both | extended}** **Example:** `switch(config-router-neighbor-af)# send-community both`	Specifies that a communities attribute should be sent to both BGP neighbors.
Step 7	**vrf** *vrf-name* **Example:** `switch(config-router-neighbor-af)# vrf VPN1`	Associates the BGP process with a VRF.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-router-vrf)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family and enters address family configuration mode.
Step 9	**exit** **Example:** `switch(config-vrf-af)# exit`	Exits IPv4 address family.
Step 10	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring InterAS Option B (with RFC 3107 implementation)



Note This implementation is the same as the option B implementation (explained in the previous task), except that BGP is used to advertise labels for loopbacks of ASBRs. In the option B (without RFC 3107) implementation, a tunnel label was advertised by IGP+LDP between ASBRs. Here, BGP is used to advertise the label and there is no need to run an IGP between the ASBRs.

Configuring the Switch for InterAS Option B (with RFC 3107 implementation), page 24-439

Configuring BGP for InterAS Option B (with RFC 3107 implementation), page 24-440

Creating an ACL to filter LDP connections between the ASBRs (RFC 3107 implementation), page 24-444

Configuring the Switch for InterAS Option B (with RFC 3107 implementation)

Configure VRFs on the DC edge switches with following steps.

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set in the default VDC. **Note** You can only install and enable MPLS in the default VDC. Use the **no** form of this command to uninstall the MPLS feature set.
Step 3	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS LDP feature on the device. **Note** When the MPLS LDP feature is disabled on the device, no LDP commands are available.

	Command	Purpose
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 5	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 6	**vrf-context** *vrf-name* **Example:** `switch(config)# vrf context VPN1`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd 100:1`	Configures the route distinguisher. The route-distinguisher argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.
Step 9	**route-target {import | export}** *route-target-ext-community* **Example:** `switch(config-vrf-af-ip4)# route-target import 1:1`	Specifies a route-target extended community for a VRF as follows: • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities.
Step 10	**copy running-config startup-config** **Example:** `switch(config-vrf-af-ip4)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring BGP for InterAS Option B (with RFC 3107 implementation)

Configure DC Edge switches with EBGP VPNv4/v6 along with BGP labeled unicast family with the following steps:

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**router bgp** *as-number* **Example:** `switch(config)# router bgp 200`	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device. In this example, the ASBR and AS are ASBR2 and AS-200, respectively.
Step 3	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-router)# address-family ipv4 unicast`	Enters address family configuration mode for configuring IPv4 or IPv6 sessions.
Step 4	**redistribute direct route-map** *tag* **Example:** `switch(config-router-af)# redistribute direct route-map loopback`	Redistributes directly connected routes using the Border Gateway Protocol.
Step 5	**allocate-label all** **Example:** `switch(config-router-af)# allocate-label all`	Configures ASBRs with the BGP labeled unicast address family to advertise labels for the connected interface.
Step 6	**exit** **Example:** `switch(config-router-af)# exit`	Exits address family router configuration mode and enters router BGP configuration mode.
Step 7	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 10.1.1.1`	Configures the BGP neighbour's IP address, and enters router BGP neighbour configuration mode.
Step 8	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 100`	Specifies the BGP neighbour's AS number.
Step 9	**address-family {ipv4|ipv6} labeled-unicast** **Example:** `switch(config-router-neighbor)# address-family ipv4 labeled-unicast`	Configures the ASBR with the *BGP labeled unicast* address family to advertise labels for the connected interface. **Note** This is the command that implements RFC 3107.

	Command	Purpose
Step 10	**exit** **Example:** `Switch(config-router-neighbor-af)# exit`	**Note** Enter the exit command twice. Exits router BGP neighbour address family configuration mode and returns to router BGP configuration mode.
Step 11	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 1.1.1.1`	Configures a loopback IP address, and enters router BGP neighbor configuration mode.
Step 12	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 100`	Specifies the BGP neighbour's AS number.
Step 13	**address-family {vpnv4|vpnv6} unicast** **Example:** `switch(config-router-neighbor)# address-family vpnv4 unicast`	Configures the ASBR with the *BGP VPNv4 unicast* address family.
Step 14	**exit** **Example:** `switch(config-router-neighbor-af)# exit`	Enters router BGP neighbor configuration mode again.
Step 15	**address-family {vpnv4|vpnv6} unicast** **Example:** `switch(config-router-neighbor)# address-family vpnv6 unicast`	Configures the ASBR with the *BGP VPNv6 unicast* address family.
Step 16	**Repeat the process with ASBR2.**	Configures ASBR2 with option B (RFC 3107) settings and implements complete IGP isolation between the two data centers DC1 and DC2.
Step 17	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configure DC Edge switches with IBGP VPNv4 with the following steps:

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal	Enters global configuration mode.
Step 2	**router bgp** *as-number* **Example:** switch(config)# router bgp 200	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device. In this example, the ASBR and AS are ASBR2 and AS-200, respectively.
Step 3	**address-family {ipv4 | ipv6} unicast** **Example:** switch(config-router)# address-family ipv4 unicast	Enters address family configuration mode for configuring IPv4 or IPv6 sessions.
Step 4	**redistribute direct route-map** *tag* **Example:** switch(config-router-af)# redistribute direct route-map loopback	Redistributes directly connected routes using the Border Gateway Protocol.
Step 5	**allocate-label all** **Example:** switch(config-router-af)# allocate-label all	Configures ASBRs with the BGP labeled unicast address family to advertise labels for the connected interface.
Step 6	**exit** **Example:** switch(config-router-af)# exit	Exits address family router configuration mode and enters router BGP configuration mode.
Step 7	**neighbor** *ip-address* **Example:** switch(config-router)# neighbor 10.1.1.1	Configures the BGP neighbour's IP address, and enters router BGP neighbour configuration mode.
Step 8	**remote-as** *as-number* **Example:** switch(config-router-neighbor)# remote-as 100	Specifies the BGP neighbour's AS number.
Step 9	**exit** **Example:** Switch(config-router-neighbor-af)# exit	**Note** Enter the exit command twice. Exits router BGP neighbour address family configuration mode and returns to router BGP configuration mode.
Step 10	**neighbor** *ip-address* **Example:** switch(config-router)# neighbor 1.1.1.1	Configures a loopback IP address, and enters router BGP neighbor configuration mode.

	Command	Purpose
Step 11	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 100`	Specifies the BGP neighbour's AS number.
Step 12	**address-family {vpnv4|vpnv6} unicast** **Example:** `switch(config-router-neighbor)# address-family vpnv4 unicast`	Configures the ASBR with the *BGP VPNv4 unicast* address family.
Step 13	**next-hop-self** **Example:** `switch(config-router-neighbor-af)# next-hop-self`	Uses the local BGP speaker address as the next-hop address in route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
Step 14	**exit** **Example:** `switch(config-router-neighbor-af)# exit`	Enters router BGP neighbor configuration mode again.
Step 15	**Repeat the process with ASBR2.**	Configures ASBR2 with option B (RFC 3107) settings and implements complete IGP isolation between the two data centers DC1 and DC2.
Step 16	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Creating an ACL to filter LDP connections between the ASBRs (RFC 3107 implementation)

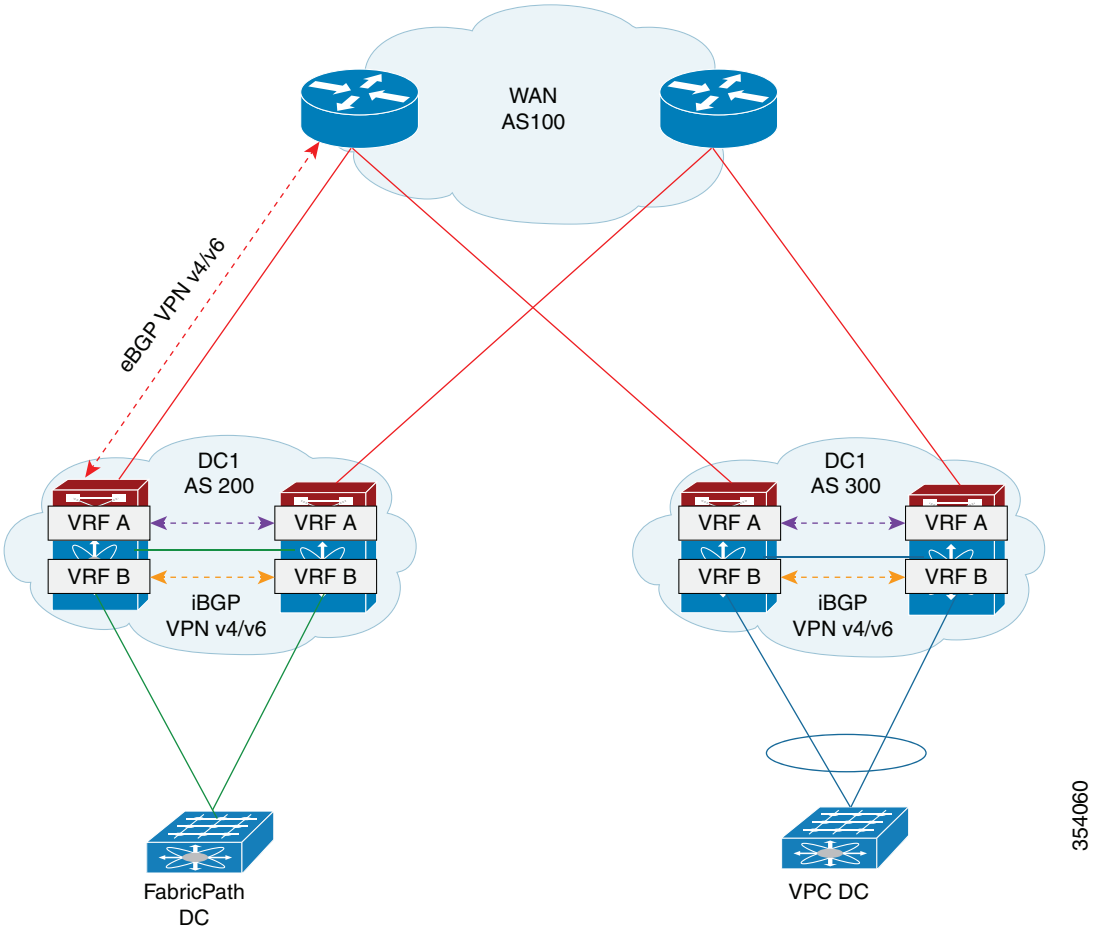
	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**ip access-list** *name* **Example:** `switch(config)# ip access-list LDP`	Creates an access list and enters ACL configuration mode.
Step 3	**[***sequence-number***] deny tcp any any eq** *packet-length* **Example:** `switch(config-acl)# 10 deny tcp any any eq 646`	Executes the ACL instruction as per the specified sequence.

	Command	Purpose
Step 4	**[***sequence-number***] deny tcp any eq** *packet-length* **any** **Example:** `switch(config-acl)# 20 deny tcp any eq 646 any`	Executes the ACL instruction as per the specified sequence.
Step 5	**[***sequence-number***] deny udp any any eq** *packet-length* **Example:** `switch(config-acl)# 30 deny udp any any eq 646`	Executes the ACL instruction as per the specified sequence.
Step 6	**[***sequence-number***] deny udp any eq** *packet-length* **any** **Example:** `switch(config-acl)# 40 deny udp any eq 646 any`	Executes the ACL instruction as per the specified sequence.
Step 7	**[***sequence-number***] permit ip any any** **Example:** `switch(config-acl)# 50 permit ip any any`	Executes the ACL instruction as per the specified sequence.
Step 8	**exit** **Example:** `switch(config-acl)# exit`	Exits ACL configuration mode and enters global configuration mode.
Step 9	**interface** *type number* **Example:** `switch(config)# interface ethernet 2/20`	Enters interface configuration mode.
Step 10	**mpls ip** **Example:** `switch(config-if)# mpls ip`	Configures MPLS hop-by-hop forwarding on this interface.
Step 11	**ip access-group** *name* **in** **Example:** `switch(config-if)# ip access-group LDP in`	Specifies that the ACL (named LDP created in the earlier steps) be applied to inbound traffic on the interface.
Step 12	**ip access-group** *name* **out** **Example:** `switch(config-if)# ip access-group LDP out`	Specifies that the ACL (named LDP created in the earlier steps) be applied to the outbound traffic on the interface.
Step 13	**end** **Example:** `switch(config-if)# end`	Exits interface configuration mode and returns to the privileged EXEC mode.

Configuring InterAS Option B (*lite* Version)

Guidelines and Limitations for Configuring InterAS Option B lite [applicable to the 6.2(2) release version]

- The aggregation switch supports only local VRFs, and Nexus devices within an autonomous system (AS) are connected through a VRF implementation.
- Routes learned from the IBGP peer are not sent to the EBGP peer and routes learned from an EBGP peer are not sent to IBGP VPNv4/VPNv6 peers.
- The interAS option B with MP-BGP on the EBGP side does not work with MP-BGP on the IBGP side. One interface goes to the core and one interface goes to the Layer 3 VPN.
- MP-BGP Layer 3 VPN does not work within an AS.



This section contains information on the following topics:

Configuring the Switch for InterAS Option B (*lite* version)

You enable certain features on the switch to run interAS option B.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

The **install feature-set mpls** command is available only in the default VDC, and you must enable it in default VDC.

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set in the default VDC. **Note** You can only install and enable MPLS in the default VDC. Use the **no** form of this command to uninstall the MPLS feature set.
Step 3	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS LDP feature on the device. **Note** When the MPLS LDP feature is disabled on the device, no LDP commands are available.
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 5	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 6	**vrf-context** *vrf-name* **Example:** `switch(config)# vrf-context VPN1`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd 100:1`	Configures the route distinguisher. The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.

	Command	Purpose
Step 9	**route-target {import | export}** *route-target-ext-community* **Example:** `switch(config-vrf-af-ip4)# route-target import 1:1`	Specifies a route-target extended community for a VRF as follows: • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities.
Step 10	**copy running-config startup-config** **Example:** `switch(config-vrf-af-ip4)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring the Interfaces for InterAS Option B (*lite* Version)

Configure DC Edge switches with vrf-lite using the following steps.

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `switch(config)# interface ethernet 2/1`	Specifies the interface to configure and enters interface configuration mode.
Step 3	**description** *description* **Example:** `switch(config-if)# description To other ASBR`	Specifies a description for the interface.
Step 4	**ip address** *prefix mask* **Example:** `switch(config-if)# ip address 10.0.0.1 255.255.255.0`	Configures IP address for interface.

	Command	Purpose
Step 5	**no shutdown** **Example:** `switch(config-if)# no shutdown`	Enables interface.
Step 6	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and enters glbal configuration mode.
Step 7	**interface** *type number* **Example:** `switch(config-if)# interface ethernet 2/2`	Specifies the interface to configure and enters interface configuration mode.
Step 8	**description** *description* **Example:** `switch(config-if)# description To CE`	Specifies a description for the interface.
Step 9	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member VPN1`	Associates a VRF with the specified interface or subinterface. The *vrf-name* argument is the name assigned to a VRF.
Step 10	**ip address** *prefix mask* **Example:** `switch(config-if)# ip address 10.0.2.1 255.255.255.0`	Configures an IP address for the interface.
Step 11	**ipv6 address** *address* **Example:** `switch(config-if)# ipv6 address 2001:DB8:1::1`	Configures an IPv6 address for interface.
Step 12	**no shutdown** **Example:** `switch(config-if)# no shutdown`	Enables the interface.
Step 13	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring BGP for InterAS Option B (*lite* Version)

Prerequisites

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

Configure EBGP VPNv4/v6 on the DC Edge switches using the following steps:

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal	Enters global configuration mode.
Step 2	**router bgp** *as-number* **Example:** switch(config)# router bgp 100	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device.
Step 3	**neighbor** *ip-address* **Example:** switch(config-router)# neighbor 10.0.0.2	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 4	**remote-as** *as-number* **Example:** switch(config-router-neighbor)# remote-as 200	The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 5	**address-family {vpnv4 | vpnv6} unicast** **Example:** switch(config-router-neighbor)# address-family vpnv4 unicast	Enters address family configuration mode for configuring IP VPN sessions.
Step 6	**send-community {both | extended}** **Example:** switch(config-router-neighbor-af)# send-community both	Specifies that a *communities* attribute should be sent to both BGP neighbors.
Step 7	**vrf** *vrf-name* **Example:** switch(config-router-neighbor-af)# vrf VPN1	Associates the BGP process with a VRF.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** switch(config-router-vrf)# address-family ipv4 unicast	Enters address family configuration mode for configuring IP VPN sessions.
Step 9	**exit** **Example:** switch(config-router-vrf-af)# exit	Exits IPv4 address family.
Step 10	**copy running-config startup-config** **Example:** switch(config-router-vrf)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Verifying InterAS Option B Configuration

To verify InterAS option B configuration information, perform one of the following tasks:

Command	Purpose
show bgp {**vpnv4** | **vpnv6**} **unicast** [*ip-prefix/length* [**neighbors** *neighbor*]] {**vrf** {*vrf-name* | **all**} | **rd** *route-distinguisher*}	Displays VPN routes from the BGP table.
show bgp ipv6 unicast [**vrf** *vrf-name*]	Displays information about BGP on a VRF for 6VPE.
show forwarding {**ip** | **ipv6**} **route vrf** *vrf-name*	Displays the IP forwarding table that is associated with a VRF. Check that the loopback addresses of the local and remote CE routers are in the routing table of the PE routers.
show {**ip** | **ipv6**} **bgp** [**vrf** *vrf-name*]	Displays information about BGP on a VRF.
show ip route [*ip-address* [*mask*]] [*protocol*] **vrf** *vrf-name*	Displays the current state of the routing table. Use the *ip-address* argument to verify that CE1 has a route to CE2. Verify the routes learned by CE1. Make sure that the route for CE2 is listed.
show {**ip** | **ipv6**} **route vrf** *vrf-name*	Displays the IP routing table that is associated with a VRF. Check that the loopback addresses of the local and remote CE routers are in the routing table of the PE routers.
show running-config bgp	Displays the running configuration for BGP.
show running-config vrf *vrf-name*	Displays the running configuration for VRFs.
show vrf *vrf-name* **interface** *if-type*	Verifies the route distinguisher (RD) and interface that are configured for the VRF.
trace *destination* [**vrf** *vrf-name*]	Discovers the routes that packets take when traveling to their destination. The **trace** command can help isolate a problem if two routers cannot communicate.

Configuration Examples for Configuring InterAS Option B

Example: Configuring InterAS Option B

```
!--Configure VRFs on the DC edge switches --!

configure terminal
    install feature-set mpls
    feature mpls ldp
    feature mpls l3vpn
    feature bgp
    vrf context VPN1
        rd 100:1
        address-family ipv4 unicast
```

```
            route-target import 1:1
            copy running-config startup-config

!--Configure DC Edge switches with IBGP & EBGP VPNv4/v6 --!

configure terminal
    router bgp 100
     neighbor 10.0.0.2
     remote-as 200
     address-family vpnv4 unicast
      send-community both
        vrf VPN1
            address-family ipv4 unicast
            exit
        copy running-config startup-config
```

Example: Configuring InterAS Option B (RFC 3107)

```
!--Configure VRFs on the DC edge switches --!

configure terminal
    install feature-set mpls
    feature mpls ldp
    feature mpls l3vpn
    feature bgp
    vrf context VPN1
     rd 100:1
     address-family ipv4 unicast
        route-target import 1:1

!--Configure DC Edge switches with EBGP VPNv4/v6 --!

configure terminal
    router bgp 200
     address-family ipv4 unicast
        redistribute direct route-map loopback
        allocate-label all
         address-family vpnv4 unicast
         address-family ipv4 labeled-unicast

        neighbor 10.1.1.1
        remote-as 100
         address-family ipv4 labeled-unicast

        neighbor 1.1.1.1

        remote-as 100
         update-source loopback0
         ebgp-multihop 2
            address-family vpnv4 unicast
             send-community both
!--Repeat the process with ASBR2. --!

!--Configure DC Edge switches with IBGP VPNv4 --!

configure terminal
    router bgp 200
    neighbor 3.3.3.3
    remote-as 200
         address-family vpnv4 unicast
            next-hop-self
            send-community both
```

```
    !--Repeat the process with ASBR2. --!
         copy running-config startup-config

!--Creating an ACL to filter LDP connection between the ASBRs (RFC 3107 implementation)--!

configure terminal
    ip access-list LDP
     10 deny tcp any any eq 646
     20 deny tcp any eq 646 any
     30 deny udp any any eq 646
     40 deny udp any eq 646 any
     50 permit ip any any
     exit
    interface ethernet 2/20
     mpls ip
     ip access-group LDP in
     ip access-group LDP out
     end
```

Additional References for Configuring InterAS Option B

This section includes the following topics:

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Interface commands	*Cisco Nexus 7000 Series NX-OS Interface Command Reference*
VRF-aware services	*Cisco Nexus 7000 Series NX-OS Unicast Routing Configuration Guide*

MIBs

MIBs	MIBs Link
• MPLS-L3VPN-STD-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/dc-os/mibs

Feature History for Configuring InterAS Option B

Table 24-1 lists the release history for this feature.

Table 24-1 Feature History for InterAS Option B

Feature Name	Releases	Feature Information
InterAS option B	7.2(0)D1(1)	This feature was enhanced with the InterAS option B and InterAS option B (with RFC 3107) provisions. An IBGP VPNv4/v6 session between DC edge routers and an EBGP VPNv4/v6 session between DC edge routers and WAN routers can be established without a per VRF iBGP session between DC edge routers. The InterAS option B (with RFC 3107) implementation ensures complete IGP isolation between the data centers and WAN.
InterAS option B lite	6.2(2)	This feature was introduced as a *lite* version.

Configuring Any Transport over MPLS

This chapter describes how to configure the Any Transport over MPLS (AToM) feature.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About Any Transport over MPLS

This section includes the following topics:

Any Transport over MPLS

Any Transport over MPLS (AToM) accommodates different types of Layer 2 packets, including Ethernet and VLAN, to enable the service provider to transport different types of traffic over the backbone and accommodate all types of customers. AToM adheres to the standards developed for transporting Layer 2 packets over MPLS. Other Layer 2 solutions are proprietary, which can limit the service provider's ability to expand the network and can force the service provider to use only one vendor's equipment. Upgrading to AToM is transparent to the customer. Because the service provider network is separate from the customer network, the service provider can upgrade to AToM without disruption of service to the customer.

The successful transmission of the Layer 2 frames between PE devices is due to the configuration of the PE devices. You can set up the connection, called a pseudowire, between the routers and specify the following information on each PE device:

- The type of Layer 2 data to be transported across the pseudowire, such as Ethernet or VLAN.
- The IP address of the loopback interface of the peer PE device, which enables PE devices to communicate.
- A unique combination of peer PE IP address and virtual circuit (VC) ID that identifies the pseudowire.

AToM encapsulates Layer 2 frames at the ingress provider edge (PE) and sends them to a corresponding PE at the other end of a pseudowire. The egress PE removes the encapsulation and sends out the Layer 2 frame.

Ethernet over MPLS

Any Transport over MPLS (AToM) supports Ethernet over MPLS (EoMPLS) in two modes: VLAN and port mode.

A VLAN is a switched network that is logically segmented by functions, project teams, or applications regardless of the physical location of users. EoMPLS allows you to connect two VLAN networks that are in different locations. You must configure the provider edge (PE) devices at each end of the MPLS backbone and add a point-to-point virtual circuit (VC). Only the two PE devices at the ingress and egress points of the MPLS backbone know about the VCs dedicated to transporting Layer 2 VLAN traffic. All other devices do not have table entries for those VCs. EoMPLS in VLAN mode transports Ethernet traffic from a source 802.1Q VLAN to a destination 802.1Q VLAN over a core MPLS network.

Port mode allows a frame coming into an interface to be packed into an MPLS packet and transported over the MPLS backbone to an egress interface. The entire Ethernet frame without the preamble or frame check sequence (FCS) is transported as a single packet. Each interface is associated with one unique pseudowire VC label.

Ethernet Remote Port Shutdown

Ethernet remote port shutdown allows a service provider edge (PE) device on the local end of an Ethernet over MPLS (EoMPLS) pseudowire to detect a remote link failure and cause the shutdown of the Ethernet port on the local customer edge (CE) device. Because the Ethernet port on the local CE device is shut down, the device does not lose data by continuously sending traffic to the failed remote link. This process is beneficial if the link is configured as a static IP route.

Estimating Packet Sizes

The following calculation helps you to determine the size of the packets that travel through the core network. You must set the maximum transmission unit (MTU) on the core-facing interfaces of the provider (P) and provider edge (PE) devices to accommodate packets of the calculated size. The MTU should be greater than or equal to the total bytes of the items in the following equation:

Core MTU > = (Edge MTU + Transport header + AToM header + (MPLS label stack * MPLS label size)), where the following definitions apply:

- The edge MTU is the MTU for customer-facing devices.
- The Transport header depends on the transport type. The table below lists the specific sizes of the headers.

Transport Type	Packet Size
Ethernet VLAN	18 bytes
Ethernet port	14 bytes

- The AToM header is 4 bytes (control word).
- The MPLS label stack size depends on the configuration of the core MPLS network:
  - AToM uses one MPLS label to identify the AToM VCs (VC label). Therefore, the minimum MPLS label stack is one for directly connected AToM PEs, which are PE devices that do not have a P router between them.
  - If the Label Distribution Protocol (LDP) is used in the MPLS network, the label stack size is two (the LDP label and the VC label).
  - If a traffic engineering (TE) tunnel is used instead of LDP between PE routers in the MPLS network, the label stack size is two (the TE label and the VC label).
  - If a TE tunnel and LDP are used in the MPLS network (for example, a TE tunnel between P routers or between P and PE routers, with LDP on the tunnel), the label stack is three (the TE label, LDP label, and VC label).
  - If you use MPLS fast reroute in the MPLS network, you add a label to the stack. The maximum MPLS label stack in this case is four (the Fast Reroute (FRR) label, TE label, LDP label, and VC label).
  - If AToM is used by the customer carrier in an MPLS VPN Carrier Supporting Carrier environment, you add a label to the stack. The maximum MPLS label stack in the provider carrier network is five (the FRR label, TE label, LDP label, VPN label, and VC label).
  - If an AToM tunnel spans different service providers that exchange MPLS labels using IPv4 Border Gateway Protocol (BGP) (RFC 3107), you add a label to the stack. The maximum MPLS label stack is five (the FRR label, TE label, LDP label, and VC label).

- Other circumstances can increase the MPLS label stack size. Therefore, analyze the complete data path between the AToM tunnel endpoints, determine the maximum MPLS label stack size for your network, and then multiply the label stack size by the size of the MPLS label.

Note For more information about establishing nondirectly connected MPLS LDP sessions, see the "Configuring MPLS Label Distribution Protocol" chapter.

Applying the following assumptions and using the formula: Edge MTU + Transport header + AToM header + (MPLS label stack * MPLS label) = Core MTU, or 1500 + 18 + 0 + (2 * 4) = 1526, you must configure the P and PE devices in the core to accept packets of 1526 bytes.

- The edge MTU is 1500 bytes.
- The transport type is Ethernet VLAN which is 18 bytes for the transport header.
- The AToM header is 0 because the control word is not used.
- The MPLS label stack is 2 because LDP is used.
- The MPLS label is 4 bytes.

Layer 2 VPN Internetworking

Layer 2 transport over Multiprotocol Label Switching (MPLS) already exists for like-to-like attachment circuits, such as Ethernet-to-Ethernet. Layer 2 Virtual Private Network (L2VPN) internetworking builds on this functionality by allowing disparate attachment circuits to be connected. The internetworking function facilitates the translation between the different Layer 2 encapsulations.

The EoMPLS L2VPN Internetworking feature supports Ethernet and VLAN attachment circuits over MPLS. The features and restrictions for like-to-like functionality also apply to L2VPN internetworking.

Quality of Service Features Supported in AToM

The table below lists the Quality of Service (QoS) features supported in AToM.

Table 25-1 QoS Features Supported in AToM

QoS Feature	EoMPLS
Service policy	Can be applied to Ethernet Virtual Circuits (EVCs) and switchport interfaces
Classification	Supports the commands for matching the following: • Class of service (CoS) on interfaces and subinterfaces • MPLS experimental topmost on interfaces and subinterfaces • QoS groups on interfaces (output policy)

QoS Feature	EoMPLS
Policing	Supports the following: • Single-rate policing • Two-rate policing • Color-aware policing • Multiple-action policing
Queuing and shaping	Supports the following: • Distributed Low Latency Queueing (dLLQ) • Distributed Weighted Random Early Detection (dWRED) • Byte-based WRED

Equal Cost Multiple Paths on PWE Label

Equal Cost Multiple Paths (ECMPs) are available between the ingress and egress devices. However, a pseudowire is transported over a single network path to retain the characteristics of the emulated service over the pseudowire.

In the network core, load balancing is performed by checking the first nibble in the frame, after the MPLS label stack. If the destination MAC address (DMAC) starts with 4 or 6, it selects a different link in the core. To avoid a different link and preserve order of frames, a control word is added to the frame transmitted over the pseudowire emulation (PWE) label.

Licensing Requirements for Any Transport over MPLS

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Layer 2 MVPN requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations for Any Transport over MPLS

Any Transport over MPLS (AToM) has the following configuration guidelines and limitations:

- Address format—Configure the Label Distribution Protocol (LDP) router ID on all PE routers to be a loopback address with a /32 mask. Otherwise, some configurations might not function properly.

Ethernet over MPLS (EoMPLS) has the following guidelines and limitations:

- EoMPLS supports VLAN packets that conform to the IEEE 802.1Q standard. The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames. The Inter-Switch Link (ISL) protocol is not supported between the PE and customer edge (CE) devices.
- EoMPLS, AToM L2VPN VC over BGP Labeled Unit may fail, as BGP Labeled Unit stitching with MPLS LDP is not currently supported.

- The AToM control word is supported. However, if the peer PE does not support a control word, the control word is disabled. This negotiation is done by LDP label binding.
- Ethernet packets with hardware-level cyclic redundancy check (CRC) errors, framing errors, and runt packets are discarded on input.
- A switch can act as the terminating provider edge (T-PE) router and peer with the subscriber provider edge (S-PE) router. But a switch cannot act as an S-PE router.
- Although you can set the MPLS maximum transmission unit (MTU) to a value less than the interface MTU, you must set the MPLS MTU to a value greater than or equal to the interface MTU to prevent data corruption, dropped packets, and high CPU rates.
- If the interface MTU is greater than or equal to 1524 bytes, you can set the maximum MPLS MTU as high as the interface MTU. For example, if the interface MTU is set to 1600 bytes, you can set the MPLS MTU to a maximum of 1600 bytes. If you set the MPLS MTU to a value higher than the interface MTU, traffic is dropped.
- For interfaces that do not allow you to configure the interface MTU value and for interfaces where the interface MTU is 1500 bytes, the MPLS MTU range is 64 to 1524 bytes.

Per-interface Ethernet over MPLS (EoMPLS) has the following guidelines and restrictions:

- The Virtual Private LAN Service (VPLS) feature does not support MTU values in pseudowire interface configuration mode.
- The device uses an MTU validation process for remote virtual circuits (VCs) established through LDP, which compares the MTU value configured in pseudowire interface configuration mode to the MTU value of the remote customer interface. If an MTU value has not been configured in pseudowire interface configuration mode, the validation process compares the MTU value of the local customer interface to the MTU value of the remote, either explicitly configured or inherited from the underlying interface or subinterface.
- When you configure the MTU value in pseudowire interface configuration mode, the specified MTU value is not enforced by the dataplane. The dataplane enforces the MTU values of the interface (port mode) or subinterface (VLAN mode).
- Ensure that the interface MTU is larger than the MTU value configured in pseudowire interface configuration mode. If the MTU value of the customer-facing subinterface is larger than the MTU value of the core-facing interface, traffic might not be able to travel across the pseudowire.

Configuring Any Transport over MPLS

This section includes the following topics:

Configuring a Pseudowire

BEFORE YOU BEGIN

Ensure that you configured the EFP (service instance) for EoMPLS. For information, see the "Configuring Ethernet over MPLS" chapter.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile type pseudowire** *profile-name*
3. **encapsulation mpls**
4. [**no**] **interface pseudowire** *pw-id*
5. (Optional) **control-word**
6. **inherit port-profile** *profile-name*
7. **neighbor** *peer-ip-address vc-id*
8. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**port-profile type pseudowire** *profile-name* **Example:** `switch(config)# port-profile type pseudowire AToM` `switch(config-if-prof)#`	Enters interface port-profile configuration mode and configures a pseudowire port profile.
Step 3	**encapsulation mpls** **Example:** `switch(config-if-prof)# encapsulation mpls`	Specifies MPLS encapsulation for this profile.
Step 4	[**no**] **interface pseudowire** *pw-id* **Example:** `switch(config-prof)# interface pseudowire 12` `switch(config-if-pseudowire)#`	Enters interface pseudowire configuration mode and configures a static pseudowire logical interface. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. **Note** You can use the **no** form of this command to delete the pseudowire interface and the associated configuration.

	Command	Purpose
Step 5	**control-word** **Example:** `switch(config-if-pseudowire)# control-word`	(Optional) Enables the control word for this interface. • If you do not enable a control word, autosense is the default mode for the control word.
Step 6	**inherit port-profile** *profile-name* **Example:** `switch(config-if-pseudowire)# inherit port-profile AToM`	Applies a port profile to this interface.
Step 7	**neighbor** *peer-ip-address vc-id* **Example:** `switch(config-if-pseudowire)# neighbor 10.2.2.1 1`	Configures a emulated virtual circuit for this interface. • The combination of the *peer-ip-address* and *vc-id* arguments must be unique on a device. • The peer IP address is the address of the provider edge (PE) peer. • The *vc-id* argument is an identifier for the virtual circuit between devices. The valid range is from 1 to 4294967295.
Step 8	**copy running-config startup-config** **Example:** `switch(config-xconnect)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Ethernet Remote Port Shutdown (optional)

The Remote Ethernet Port Shutdown feature is enabled by default when an image with the feature supported is loaded on the device.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **l2vpn xconnect context** *context-name*
3. [**no**] **remote failure notification**
4. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **l2vpn xconnect context** *context-name* **Example:** `switch(config)# l2vpn context cxt1` `switch(config-xconnect)#`	Enters Xconnect configuration mode and establishes a Layer 2 VPN (L2VPN) context for identifying the two members in a VPWS, multisegment pseudowire, or local connect service. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 3	[**no**] **remote failure notification** **Example:** `switch(config-xconnect)# remote failure notification`	Enables AToM MPLS remote link failure notification and shutdown. **Note** You can use the **no** form of this command to disable this feature.
Step 4	**copy running-config startup-config** **Example:** `switch(config-xconnect)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Ethernet over MPLS in VLAN Mode

You can configure EoMPLS (VLAN mode) on the subinterfaces.

BEFORE YOU BEGIN

- Ensure that you configured the EFP (service instance) for EoMPLS. For information, see the "Configuring Ethernet over MPLS" chapter.
- Before configuring Ethernet over MPLS (EoMPLS) in VLAN mode, you must configure EoMPLS on the subinterfaces.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/subslot/port*[**.***subinterface*]
3. **encapsulation dot1q** *vlan-id*
4. [**no**] **l2vpn context** *context-name*

5. (Optional) **internetworking** {**ethernet** | **vlan**}
6. [**no**] **member** *interface-type slot/port* [**service-instance** *service-instance-id*] [**group** *group-name*] [**priority** *number*]
7. [**no**] **member pseudowire** *pw-id* [**group** *name*] [**priority** *number*]
8. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface ethernet** *slot/subslot/port*[**.***subinterface*] **Example:** `switch(config)# interface ethernet 4/0/0.1` `switch(config-if)#`	Enters interface configuration mode. • Ensure the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 3	**encapsulation dot1q** *vlan-id* **Example:** switch(config-if)# encapsulation dot1q 100	Configures the matching criteria for mapping dot1q frames on an ingress interface to this interface. • The valid range for the *vlan-id* argument is from 2 to 967. • The subinterfaces between the CE and PE routers that are running EoMPLS must be in the same subnet. All other subinterfaces and backbone devices do not ned to be in the same subnet.
Step 4	[**no**] **l2vpn xconnect context** *context-name* **Example:** `switch(config-if)# l2vpn context cxt1` `switch(config-xconnect)#`	Enters XConnect configuration mode and establishes a Layer 2 VPN (L2VPN) context for identifying the two members in a VPWS, multisegment pseudowire, or local connect service. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.

	Command	Purpose
Step 5	**internetworking** {**ethernet** | **vlan**} **Example:** `switch(config-xconnect)# internetworking ethernet`	(Optional) Specifies the type of pseudowire and the type of traffic that can flow across it. • This command is required only if you are configuring a connection between two disparate attachment circuits. • The internetworking type on a provider edge (PE) device must match the internetworking type on its peer PE device. • The **ethernet** keyword causes Ethernet frames to be extracted from the attachment circuit and sent over the pseudowire. Ethernet end-to-end transmission is assumed. Attachment circuit frames that are not Ethernet are dropped. • The **vlan** keyword allows the VLAN ID to be included as part of the Ethernet frame.
Step 6	[**no**] **member** *interface-type slot*/*port* [**service-instance** *service-instance-id*] [**group** *group-name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member ethernet 0/0/0.1 service-instance 300`	Adds an active Ethernet AC, with or without an Ethernet Flow Point (EFP), to the context. • The *service-instance-id* argument is a unique per-interface identifier for the EFP. The valid range is from 1 to 4000. The range might be restricted due to resource constraints. • (Optional) The **group** *group-name* keyword and argument combination specifies to which of the redundant groups the member belongs. This configuration is required if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. • You can use the **no** form of this command to delete the specified member configuration.

	Command	Purpose
Step 7	[**no**] **member pseudowire** pw-id [**group** group-*name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member pseudowire 12 group core-side priority 1`	Adds an active pseudowire to the context. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. • (Optional) The **group** *group-name* keyword and argument combination specifies to which of the redundant groups the member belongs. This configuration is required if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. • You can use the **no** form of this command to delete the specified member configuration.
Step 8	**copy running-config startup-config** **Example:** `switch(config-xconnect)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Ethernet over MPLS in Port Mode

Perform this task to configure EoMPLS (port mode) on the subinterfaces.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/subslot/port*[**.***subinterface*]
3. **l2vpn xconnect context** *context-name*
4. [**no**] **member** *interface-type slot/port* [**service-instance** *service-instance-id*] [**group** *group-name*] [**priority** *number*]
5. [**no**] **member pseudowire** *pw-id* [**group** *name*] [**priority** *number*]
6. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface ethernet** *slot/subslot/port*[**.***subinterface*] **Example:** `switch(config)# interface ethernet 4/0/0` `switch(config-if)#`	Enters interface configuration mode. • Ensure the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 3	[**no**] **l2vpn xconnect context** *context-name* **Example:** `switch(config-if)# l2vpn context cxt1` `switch(config-xconnect)#`	Enters XConnect configuration mode and establishes a Layer 2 VPN (L2VPN) context for identifying the two members in a VPWS, multisegment pseudowire, or local connect service. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 4	[**no**] **member** *interface-type slot/port* [**service-instance** *service-instance-id*] [**group** *group-name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member ethernet 0/0`	Adds an active Ethernet AC, with or without an Ethernet Flow Point (EFP), to the context. • The *service-instance-id* argument is a unique per-interface identifier for the EFP. The valid range is from 1 to 4000. The range might be restricted due to resource constraints. • (Optional) The **group** keyword specifies which of redundant groups the member belongs. This must be configured if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. • You can use the **no** form of this command to delete the specified member configuration.

	Command	Purpose
Step 5	[**no**] **member pseudowire** pw-id [**group** *name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member pseudowire 12`	Adds an active pseudowire to the context. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. • (Optional) The **group** keyword specifies which of redundant groups the member belongs. This must be configured if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. • You can use the **no** form of this command to delete the specified member configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-xconnect)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Per-Subinterface MTU for Ethernet over MPLS

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **mtu** *mtu-value*
4. **encapsulation dot1q** *vlan-id*
5. [**no**] **l2vpn context** *context-name* **encapsulation mpls**
6. **mtu** *mtu-value*
7. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface ethernet** *slot/port* **Example:** `switch(config)# interface ethernet 3/1` `switch(config-if)#`	Enters interface configuration mode.
Step 3	**mtu** *mtu-value* **Example:** `switch(config-if)# mtu 2000`	Configures the maximum transmission unit (MTU) size, in bytes, for this interface. • The valid range for the *mtu-value* argument is 576 to 9216. The default is 1500.
Step 4	**encapsulation dot1q** *vlan-id* **Example:** switch(config-if)# encapsulation dot1q 100	Configures the matching criteria for mapping dot1q frames on an ingress interface to this EFP. • The valid range for the *vlan-id* argument is from 2 to 967. • The subinterfaces between the CE and PE routers that are running EoMPLS must be in the same subnet. All other subinterfaces and backbone devices do not need to be in the same subnet.
Step 5	[**no**] **l2vpn context** *context-name* **encapsulation mpls** **Example:** `switch(config-if)# l2vpn context cxt1 encapsulation mpls` `switch(config-xconnect)#`	Enters Xconnect configuration mode and establishes a Layer 2 VPN (L2VPN) context for identifying the two members in a VPWS, multisegment pseudowire, or local connect service. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. • The **encapsulation** and **mpls** keywords specify MPLS encapsulation for this context. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 6	**mtu** *mtu-value* **Example:** `switch(config-xconnect)# mtu 1400`	Configures the maximum transmission unit (MTU) size, in bytes, for this context. • The valid range for the *mtu-value* argument is 576 to 9216. The default is 1500.
Step 7	**copy running-config startup-config** **Example:** `switch(config-xconnect)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying Any Transport over MPLS

To verify configuration information, perform one of the following tasks:

Command	Purpose
show l2vpn atom vc detail	Displays detailed information about Any Transport over MPLS (AToM) virtual circuits (VCs) and static pseudowires that have been enabled to route Layer 2 packets on a device.
show l2vpn mpls transport binding	Displays the MTU values assigned to the local and remote interfaces.

Configuration Examples for Any Transport over MPLS

This section includes the following topics:

Example: Remote Ethernet Port Shutdown

The following example shows how to enable a remote Ethernet port shutdown:

```
interface pseudowire 100
    encapsulation mpls
    neighbor 10.1.1.1 1
!
l2vpn xconnect context con1
    remote link failure notification
```

The following example shows how to disable a remote Ethernet port shutdown:

```
interface GigabitEthernet1/0/0
    interface pseudowire 100
    encapsulation mpls
    neighbor 10.1.1.1 1
!
l2vpn xconnect context con1
    no remote link failure notification
```

Example: Configuring per-Subinterface MTU for Ethernet over MPLS

This example shows a configuration that enables matching MTU values between VC endpoints. PE1 is configured in the XConnect subinterface configuration mode with an MTU value of 1500 bytes in order to establish an end-to-end VC with PE2, which also has an MTU value of 1500 bytes.

CE1 Configuration

```
interface gigabitethernet 0/0
  mtu 1500
  no ip address
```

```
!
interface gigabitethernet 0/0.1
  encapsulation dot1Q 100
  ip address 10.181.182.1 255.255.255.0
```

PE1 Configuration

```
interface gigabitethernet 0/0
  mtu 2000
  no ip address
!
interface gigabitethernet 0/0.1
  encapsulation dot1Q 100
!
interface pseudowire 100
  neighbor 10.1.1.152 100
  encapsulation mpls
  mtu 2000
!
l2vpn xconnect context ctx1
  member gigabitethernet0/0.1
  member pseudowire 100
!
interface gigabitethernet 0/0.2
  encapsulation dot1Q 200
  ip address 10.151.100.1 255.255.255.0
  mpls ip
```

PE2 Configuration

```
interface gigabitethernet 1/0
  mtu 2000
  no ip address
!
interface gigabitethernet 1/0.2
  encapsulation dot1Q 200
  ip address 10.100.152.2 255.255.255.0
  mpls ip
!
interface fastethernet 0/0
  no ip address
!
interface fastethernet 0/0.1
  description default MTU of 1500 for FastEthernet
  encapsulation dot1Q 100
  xconnect 10.1.1.151 100 encapsulation mpls
```

CE2 Configuration

```
interface fastethernet 0/0
  no ip address
  interface fastethernet 0/0.1
  encapsulation dot1Q 100
  ip address 10.181.182.2 255.255.255.0
```

Example: Configuring MTU for Interworking

The following example shows an L2VPN interworking example. The PE1 device has a serial interface configured with an MTU value of 1492 bytes. The PE2 router is configured with a matching MTU of 1492 bytes, which allows the two devices to form an interworking VC. If the PE2 device was not explicitly configured with a matching MTU value, the interface would be set to 1500 bytes by default and the VC would not come up.

PE1 Configuration

```
interface Loopback0
  ip address 10.1.1.151 255.255.255.255
!
interface pseudowire100
  neighbor 10.1.1.152 100
  encapsulation mpls
  mtu 2000
  l2vpn xconnect context ctx1
  member gigabitethernet0/0
  member pseudowire 100
!
router ospf 1
  log-adjacency-changes
  network 10.1.1.151 0.0.0.0 area 0
  network 10.151.100.0 0.0.0.3 area 0
!
mpls ldp router-id Loopback0
```

PE2 Configuration

```
pseudowire-class atom-ipiw
  encapsulation mpls
  interworking ip
!
interface Loopback0
  ip address 10.1.1.152 255.255.255.255
!
interface Ethernet0/0
  no ip address
  xconnect 10.1.1.151 123 pw-class atom-ipiw
    mtu 1492
!
interface Serial4/0
  ip address 10.100.152.2 255.255.255.252
  encapsulation ppp
  mpls ip
  serial restart-delay 0
!
router ospf 1
  log-adjacency-changes
  network 10.1.1.152 0.0.0.0 area 0
  network 10.100.152.0 0.0.0.3 area 0
!
mpls ldp router-id Loopback0
```

Additional References for Any Transport over MPLS

For additional information about provisioning static pseudowires for Any Transport over MPLS (AToM), see the following section:

- Related Documents, page 25-474

Related Documents

Related Topic	Document Title
Interface commands	*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*
VLAN commands	Cisco Nexus 7000 Series NX-OS Layer 2 Switching Command Reference
Ethernet over MPLS	"Configuring Ethernet over MPLS" chapter
Non directly connected MPLS LDP sessions	"Configuring the MPLS Label Distribution Protocol" chapter

Feature Information for Any Transport over MPLS

Table 25-2 lists the release history for this feature.

Table 25-2 Feature Information for Any Transport over MPLS

Feature Name	Releases	Feature Information
Any Transport over MPLS	6.2(2)	The Any Transport over MPLS (AToM) feature provides the following capabilities: • Transports data link layer (Layer2) packets over a Multiprotocol Label Switching (MPLS) backbone. • Enables service providers to connect customer sites with existing Layer 2 networks by using a single, integrated, packet-based network infrastructure—a Cisco MPLS network. Instead of using separate networks with network management environments, service providers can deliver Layer 2 connections over an MPLS backbone. • Provides a common framework to encapsulate and transport Ethernet traffic over an MPLS network core.

Configuring Any Transport over MPLS Pseudowire Provisioning

This chapter describes how to configure static pseudowires in cases where you cannot use directed control protocols, such as Label Distribution Protocol or Resource Reservation Protocol over traffic-engineered (RSVP-TE) tunnels.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the “New and Changed Information” chapter or the Feature History table below.

Licensing Requirements for Any Transport over MPLS Pseudowire Provisioning

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Layer 2 MVPN requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations for Any Transport over MPLS Pseudowire Provisioning

Any Transport over MPLS (AToM) pseudowire provisioning has the following guidelines and limitations:

- A traffic engineering (TE) tunnel can be configured and the Interior Gateway Protocol (IGP) can use the tunnel as the outgoing route for pseudowire packets. For more information, see the "Configuring the Basic MPLS TE" chapter.
- The following functionality is not supported for static pseudowires:
  - Tunnel stitching is not supported.
  - Pseudowire redundancy is not supported.
  - Autosensing of the virtual circuit type for Ethernet over MPLS (EoMPLS) is not supported.

Configuring Any Transport over MPLS Pseudowire Provisioning

SUMMARY STEPS

1. **configure terminal**
2. **mpls label range** *min-label max-label* [**static** *min-static-label max-static-label*]
3. [**no**] **interface pseudowire** *pw-id*
4. **signaling protocol none**
5. **neighbor** *peer-ip-address vc-id*
6. **encapsulation mpls**
7. **label** *local remote*
8. **no shutdown**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls label range** *min-label max-label* [**static** *min-static-label max-static-label*] **Example:** `switch(config)# mpls label range 20 199 static 200 400`	Reserves a range of labels for static label assignments. • The values for the *min-label* and *max-label* arguments identify a generic group of reserved label numbers. • The optional **static** keyword reserves a group of label numbers specifically for assigning static pseudowires labels. • The range is from 16 to 471804 for all arguments.
Step 3	[**no**] **interface pseudowire** *pw-id* **Example:** `switch(config)# interface pseudowire 12` `switch(config-if-pseudowire)#`	Enters interface pseudowire configuration mode and configures a static pseudowire logical interface. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. **Note** You can use the **no** form of this command to delete the pseudowire interface and the associated configuration.
Step 4	**signaling protocol none** **Example:** `switch(config-if-pseudowire)# signaling protocol none`	Disables signaling to use manually configured pseudowires.
Step 5	**neighbor** *peer-ip-address vc-id* **Example:** `switch(config-if-pseudowire)# neighbor 10.2.2.2 100`	Configures a emulated virtual circuit for this interface. • The combination of the *peer-ip-address* and *vc-id* arguments must be unique on a device. • The peer IP address is the address of the provider edge (PE) peer. • The *vc-id* argument is an identifier for the virtual circuit between devices. The valid range is from 1 to 4294967295.
Step 6	**encapsulation mpls** **Example:** `switch(config-if-pseudowire)# encapsulation mpls` `switch(config-pseudowire-mpls)#`	Specifies MPLS encapsulation for this profile.

	Command	Purpose
Step 7	**label** *local remote* **Example:** `switch(config-pseudowire-mpls)# label 200 400`	Sets a value for the local pseudowire label and the remote pseudowire label. • The value for the *local* and *remote* arguments must be within the range specified by the **mpls label range** command with the **static** keyword.
Step 8	**no shutdown** **Example:** `switch(config-pseudowire-mpls)# no shutdown`	Brings the port administratively up.
Step 9	**copy running-config startup-config** **Example:** `switch(config-pseudowire-mpls)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying Any Transport over MPLS Pseudowire Provisioning

To verify pseudowire configuration information, perform the following task:

Command	Purpose
show l2vpn atom vc detail	Displays detailed information about Any Transport over MPLS (AToM) virtual circuits (VCs) and static pseudowires that have been enabled to route Layer 2 packets on a device.

Additional References for Any Transport over MPLS Pseudowire Provisioning

For additional information about provisioning static pseudowires for Any Transport over MPLS (AToM), see the following section:

• Related Documents, page 26-480

Related Documents

Related Topic	Document Title
Interface commands	*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*
VLAN commands	Cisco Nexus 7000 Series NX-OS Layer 2 Switching Command Reference
Traffic engineering tunneling	"Configuring the Basic MPLS TE" chapter

Feature Information for Any Transport over MPLS Pseudowire Provisioning

Table 26-1 lists the release history for this feature.

Table 26-1 Feature Information for Any Transport over MPLS Pseudowire Provisioning

Feature Name	Releases	Feature Information
Any Transport over MPLS Pseudowire Provisioning	6.2(2)	The Any Transport over MPLS (AToM) Pseudowire Provisioning feature allows you to provision an AToM pseudowire without the use of a directed control connection. In environments that do not or cannot use directed control protocols, this feature provides a means for provisioning the pseudowire parameters statically at the command-line interface.

Configuring Ethernet over MPLS

Note The Cisco NX-OS release that is running on a managed device may not support all the features or settings described in this chapter. For the latest feature information and caveats, see the documentation and release notes for your platform and software release.

This chapter describes how to configure Ethernet Virtual Circuits (EVCs) using the Cisco Data Center Network Manager (DCNM) for Ethernet over Multiprotocol Label Switching (EoMPLS) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the “New and Changed Information” chapter or the Feature History table below.

Information About Ethernet over MPLS

This section includes the following topics:

Layer 2 Services

A Layer 2 Virtual Private Network (L2VPN) enables service providers to carry multiple network services over a single converged network using Multiprotocol Label Switching (MPLS). MPLS L2VPN extends the Layer 2 domains in data centers. MPLS can be used to connect branch offices to back up data centers and also to interconnect multiple data centers in the same organization.

L2VPN services using the MPLS/IP core can be divided into two categories: wire and LAN services. The Virtual Private Wire Service (VPWS) provides point-to-point service between two customer edge (CE) devices over the provider core. The Virtual Private LAN Service (VPLS) provides point-to-multipoint service between multiple customer sites using a mesh of point-to-point pseudowires over the provider core to emulate a LAN between the sites.

Ethernet over MPLS

Ethernet over MPLS (EoMPLS) is a VPWS service that is used to carry Layer 2 Ethernet frames over an MPLS network. EoMPLS enables service providers to offer emulated Ethernet services over existing MPLS networks.

EoMPLS encapsulates Ethernet frames in MPLS packets and forwards them across the MPLS network. Each frame is transported as a single packet, and the PE routers connected to the backbone add and remove labels as appropriate for packet encapsulation. Using EoMPLS, Layer 2 networks that are geographically separated can be connected without requiring bridges or routers at the remote locations.

Attachment Circuits

A Layer 2 circuit that connects a customer edge (CE) node to a provider edge (PE) node is known as an attachment circuit or AC. A Layer 2 VPN (L2VPN) supports only Ethernet ACs on Cisco NX-OS devices.

To cross the network core, the Layer 2 traffic is tunneled inside a pseudowire. A pseudowire is typically a Multiprotocol Label Switching (MPLS) label-switched path (LSP), or a Layer 2 Tunneling Protocol (L2TP) tunnel, or the psuedowire can be locally switched from another AC. Layer 2 VPN connects different types of circuits (that is, different types of Layer 2 ACs and pseudowires) together in different ways to implement different types of end-to-end services.

The following types of ACs are supported:

- Ethernet port mode—This AC includes all frames that are sent and received on a physical Ethernet port.
- Ethernet 802.1Q—This AC includes all frames that are sent and received with a particular VLAN tag.
- Ethernet 802.1ad (Q-in-Q)—This AC includes all frames that are sent and received with a specific outer VLAN tag and a specific inner VLAN tag. VLAN-in-VLAN (Q-in-Q) is supported only in the service instance configuration and not in the subinterface configuration.
- Ethernet QinAny—This AC includes all frames that are sent and received with a specific outer VLAN tag and any inner VLAN tags, as long as the inner VLAN tag is not used on another subinterface.

Ethernet Virtual Circuits

An Ethernet Virtual Circuit (EVC) as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint Layer 2 circuit. It is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer.

Bridge Domain

A bridge domain is a generic object that represents a Layer 2 broadcast domain on a device. The EVC architecture uses a bridge domain to define a Layer 2 multipoint service.

Creating a bridge domain also creates the underlying VLAN, if it does not already exist. There is a one-to-one mapping of bridge domains to VLANs; bridge domain 100 maps to VLAN 100.

Ethernet Flow Point

An Ethernet Flow Point (EFP) is the instantiation of an EVC on a specific interface on a device. The EFP interface representation is similar to that of a subinterface that maintains the parent-child relationship with the port.

The EFP interface is a Layer 2 logical interface. Any Layer 2 feature, protocol, or application that functions on a switchport is equally applicable to an EFP, although some constraints might apply. Similar to a physical port, the interface state machine and forwarding behavior for the EFP depends on the service to which it belongs.

An EFP interface, also known as a service instance, is implicitly created when you configure an Ethernet service instance on a port. An EFP can be configured under a physical or logical parent port. Each service instance has its own configuration submode. Different features that apply to the service instance can be configured in that submode.

Because a single parent port can support multiple service instances, several EFPs can be associated with the port, with each EFP as part of a different EVC. For this reason, whenever a service instance is configured on a port, the port is internally brought up in trunk mode.

Note The EVC represents a bridge domain. An EFP is an instance of an Ethernet flow on a particular interface that belongs to a bridge domain. The Ethernet flow, not the entire port, belongs to the bridge domain.

Flow per EFP

A trunk is a point-to-point link between the switch and another networking device. Trunks carry the traffic of multiple VLANs over a single link and allow you to extend VLANs across an entire network. EVCs can identify flows based on multiple criteria in the Layer 2 header. In Cisco NX-OS, the flow identification for devices with Enhanced Logic recognition Logic 8 (Earl8) hardware is based on matching the VLAN tag of the incoming packet. If the incoming packet has multiple VLAN tags, only the outer tag is used for traffic mapping to an EFP.

To correctly deliver the traffic on a trunk port with several VLANs, the device uses the IEEE 802.1Q encapsulation, or tagging method, that uses a tag that is inserted into the frame header. This tag carries information about the specific VLAN to which the frame and packet belong. This method allows packets that are encapsulated for several different VLANs to traverse the same port and maintain traffic separation between the VLANs. Also, the encapsulated VLAN tag allows the trunk to move traffic end-to-end through the network on the same VLAN.

Encapsulation defines the matching criteria that maps a VLAN to the service instance. A single VLAN ID can be configured for an exact match of the outermost tag. Any VLAN ID that is not specifically configured on an EFP or subinterface is treated as if it is implicitly configured for default encapsulation. On a parent port, you can configure either a single default EFP or one or more EFPs with explicit encapsulation, but not both.

VLAN Translations

VLAN translation, also know as a rewrite operation, provides flexibility in managing virtual LANs (VLANs) and Metro Ethernet-related services. VLAN translation is supported for Ethernet interfaces only, not for other types of interfaces.

With 1:1 VLAN translation, the VLAN of the incoming traffic (CE VLAN) is rewritten (replaced) by a provider edge (PE) VLAN. This process enables the service provider to address the situation where incoming traffic from two different customers share the same customer edge (CE) VLAN. The service provider can map the two CE VLANs to two different PE VLANs, and customer traffic will not be mixed.

Devices can also push (add) or pop (remove) VLAN tags in frame headers. A VLAN tag push is supported only on a service instance that is configured as the default EFP. In order to push a VLAN tag, the port mode is implicitly changed to a tunnel, making the port unable to distinguish the incoming flow based on VLAN tags. A VLAN tag is pushed for every frame that enters the port, irrespective of the incoming VLAN tag. In port mode, a VLAN tag is pushed at ingress and the same tag is popped at the egress to apply symmetric rewrites on an EFP.

Layer 2 VPN Internetworking

Layer 2 transport over Multiprotocol Label Switching (MPLS) already exists for like-to-like attachment circuits, such as Ethernet-to-Ethernet. Layer 2 Virtual Private Network (L2VPN) internetworking builds on this functionality by allowing disparate attachment circuits to be connected. The internetworking function facilitates the translation between the different Layer 2 encapsulations.

The EoMPLS L2VPN Internetworking feature supports Ethernet and VLAN attachment circuits over MPLS. The features and restrictions for like-to-like functionality also apply to L2VPN internetworking.

For more information, see the “Configuring Any Transport over MPLS” chapter.

Layer 2 VPN Stateful High Availability

The L2VPN Stateful High Availability (HA) feature uses two supervisor modules to provide uninterrupted service during a system failure. This implementation is the same for both Ethernet over Multiprotocol Label Switching (EoMPLS) and Virtual Private LAN Service (VPLS). During a failure, when an active supervisor is down, the standby supervisor seamlessly takes over all operations without disruptions. The supervisor modules also use Nonstop Forwarding (NSF), Stateful Switchover (SSO), and Graceful Restart (GR) for Any Transport over MPLS (AToM) to recover from an interruption in the service.

Peer Label Switch Routers (LSRs) exchange label binding information in an Multiprotocol Label Switching (MPLS) network to support the forwarding process. The MPLS Label Distribution Protocol Graceful Restart feature provides a mechanism by which the forwarding state between LSRs can be maintained during interruptions such as SSO failover events and temporary loss of Label Distribution Protocol (LDP) communication between the LSRs to enable NSF for MPLS traffic.

To enable NSF for Any Transport over MPLS (AToM) traffic, the provider edge (PE) devices and the LDP peers involved in the SSO event must support GR. There is no specific configuration required for Layer 2 VPN stateful HA.

Ethernet over MPLS Coexistence

This section includes the following topics. There are no specific tasks required to configure these features:

- Ethernet over Multiprotocol Label Switching and Virtual Private LAN Service, page 27-486
- Ethernet over MPLS and Cisco Overlay Transport Virtualization, page 27-487
- Ethernet over MPLS and Virtual Private Lan Service, page 27-487

Ethernet over Multiprotocol Label Switching and Virtual Private LAN Service

Ethernet over Multiprotocol Label Switching (EoMPLS) and Virtual Private LAN Service (VPLS) can coexist with MPLS Layer 3 VPNs (L3VPNs) on the same device. When you use a routed pseudowire, EoMPLS is configured on a bridge domain or a VLAN and the switched virtual interface (SVI) for that VLAN participates in Layer 3 forwarding or is part of a virtual routing and forwarding (VRF) instance. You can also configure a routed pseudowire by adding the pseudowire directly to a bridge domain. Adding a pseudowire directly to a bridge domain is not supported on all Cisco Nexus platforms. If this configuration is not supported, and VPLS is already configured on the device, the Layer 3 configuration on the SVI is rejected and vice versa.

Pseudowires enable payloads to be transparently carried across IP or MPLS packet-switched networks (PSNs). This functionality provides a Layer 3 virtual interface representation of a pseudowire on a provider edge (PE) device. This functionality also allows the backhaul of customer packets over pseudowires and the application of Layer 3 features, such as quality of service (QoS) policing and shaping, and access lists on customer packets.

A pseudowire head end allows a pseudowire to be terminated on a VRF instance; however, this termination is not required for EoMPLS co-existence.

Ethernet over MPLS and Cisco Overlay Transport Virtualization

If Ethernet over Multiprotocol Label Switching (EoMPLS) and Cisco Overlay Transport Virtualization (OTV) are configured on different bridge domains or VLANs, they can coexist on the same device. If EoMPLS and OTV coexist on a device, one part of the network uses MPLS EoMPLS and the other part uses OTV. The Provider Edge Gateway (PE-G) forwards packets between these two parts of the network. The IP (OTV) cloud and the MPLS cloud can be the same physical network. In OTV, MAC-address learning occurs in the control plane and in EoMPLS, it occurs in the data plane.

Ethernet over MPLS and Virtual Private Lan Service

Ethernet over Multiprotocol Label Switching (EoMPLS) can coexist with Virtual Private LAN Service (VPLS) on the same device because EoMPLS is configured on interfaces and VPLS is configured on a bridge domain. Point-to-point EoMPLS and VPLS can also coexist on the same device.

In EoMPLS port-mode operation, an attachment circuit (AC) cannot be part of a bridge domain of a VPLS because all incoming tagged packets are tunneled through a pseudowire. In EoMPLS VLAN mode operation, a packet with a matching VLAN is sent over a point-to-point pseudowire. Packets with other VLANs are mapped to a bridge domain and hence, these packets can participate in VPLS forwarding.

LinkSec

The LinkSec feature provides security for data centers over pseudowires using point-to-point encryption. LinkSec supports IEEE 802.1AE link-layer cryptography that provides hop-by-hop security of data in the MAC layer. Link-layer cryptography helps ensure end-to-end data privacy while enabling the insertion of security service devices along the encrypted path.

Hop-by-Hop Encryption

In this type of deployment, data is encrypted on the egress interface of the device and decrypted on the ingress interface of the device. Data is encrypted while being transmitted on interfaces but decrypted inside devices. However, if LinkSec is unavailable on certain segments of the network, data is sent in decrypted state on these segments. The advantage of this type of deployment is that Layer 2 Virtual Private Network (L2VPN) or Multiprotocol Label Switching (MPLS) is not aware of the encryption.

Hop-by-hop encryption is the default mode of encryption in LinkSec.

Encryption and Decryption at Customer Edge Devices

After Layer 2 Virtual Private Network (L2VPN) or Multiprotocol Label Switching (MPLS) has added its label information to the frame, LinkSec encrypts both the data packet and the VLAN tag. The VLAN tag is lost and LinkSec sends the entire package across the network as payload. In this type of deployment, data is encrypted and decrypted at customer edge (CE) devices only.

To enable this deployment, you should configure the provider edge (PE) ports in the port mode of L2VPN operation because the VLAN tag is lost during LinkSec encryption.

This method can also be deployed by configuring the PE ports as access switchports and mapping the packets that enter the ingress PE1 interface to an access VLAN. The packets are then forwarded using Virtual Private Lan Service (VPLS) or Ethernet over Multiprotocol Label Switching (EoMPLS) if the egress PE1 interface is configured to be part of a bridge domain of the VLAN.

MPLS Quality of Service

To maintain the quality of service (QoS) when a packet traverses both Layer 2 and Layer 3 domains, the type of service (ToS) and CoS values must be mapped to each other. CoS refers to three bits in either an Inter-Switch Link (ISL) header or an 802.1Q header that are used to indicate the priority of an Ethernet frame as it passes through a switched network.

The 802.1Q provides QoS-based matching and marking to VLAN user priority bits to provide QoS on the Gigabit Ethernet WAN interface for 802.1Q packets. Packet marking helps identify packet flows. Packet marking enables the partitioning of a network into multiple priority levels or CoS. During network congestion, packets that are marked as priority are offered a higher priority than other packets.

802.1Q input packets are classified at eight different QoS levels (0 to 7) based on the VLAN user priority bits. For 802.1Q output packets, QoS marking is done at the VLAN header to modify VLAN user priority bits. QoS services use these priority bit settings to gain traffic priority during network congestion.

Experimental Bits

EXP is a 3-bit field and part of a Multiprotocol Label Switching (MPLS) header. Experimental (EXP) bits in an MPLS header carry the priority of packets. Each label switching device along the network path honors the packet priority by queuing packets in the proper queue and servicing packets according to the priority. EXP bits define the quality of service (QoS) treatment (per-hop behavior) that a node should give to a packet. It is the equivalent of the differentiated service code point (DSCP) in the IP network. A DSCP defines a class and drop precedence. The EXP bits generally carry all information encoded in IP DSCP. However, in some cases, the EXP bits are used exclusively to encode the dropping precedence.

QoS on a Layer 2 VPN (L2VPN) network usually has two parts, an attachment circuit (AC) side and a pseudowire side. Layer 2 QoS is applied on the AC side and Layer 3 MPLS or IP QoS is applied on the pseudowire side.

Virtual Private LAN Service (VPLS) QoS is similar to Ethernet over MPLS (EoMPLS) QoS, except that QoS in VPLS is applied at ACs that participate in a VPLS bridge domain.

The core-facing MPLS interface must support a QoS policy. This QoS policy is applied on Ethernet Virtual Circuits (EVCs) and switchport interfaces. If a switchport interface participates in QoS handling, the matching criteria must include the VLAN on which VPLS forwarding is configured.

Setting the EXP bit value helps service providers who do not want to modify the value of the IP precedence field within the IP packets that are transported through their networks. By choosing different values for the Multiprotocol Label Switching (MPLS) EXP bit field, you can specify the priority that a packet requires during periods of network congestion. By default, the IP precedence value is copied into

the MPLS EXP field during imposition. On the imposition path, packets are received from the AC and are sent to the MPLS core. You can specify the MPLS EXP bits with an MPLS quality of service (QoS) policy.

By default, EXP is derived from COS for VPLS and VLAN-based EoMPLS. For port-based EoMPLS, by default, EXP is derived from the DSCP value.

Licensing Requirements for Ethernet over MPLS

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco DCNM	IP tunnels require a LAN Enterprise license. For a complete explanation of the DCNM licensing scheme and how to obtain and apply licenses, see the *Cisco DCNM Installation and Licensing Guide, Release 5.x.*
Cisco NX-OS	Layer 2 MVPNs require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites

- MPLS must be configured in the core so that a label-switched path (LSP) exists between the provider edge (PE) devices.
- Cisco Express Forwarding must be enabled before you configure any Layer 2 circuits.

Guidelines and Limitations for Ethernet over MPLS

Ethernet over MPLS (EoMPLS) has the following guidelines and limitations:

- EoMPLS, AToM L2VPN VC over BGP Labeled Unit may fail, as BGP Labeled Unit stitching with MPLS LDP is not currently supported.
- Fabric Extender (FEX) ports are not supported as members of either XConnect or virtual forwarding instance (VFI) contexts.
- EoMPLS and VPLS can coexist on the same device.
- EoMPLS and VPLS can coexist with MPLS Layer 3 VPNs on the same device.
- If EoMPLS and Cisco Overlay Transport Virtualization (OTV) are configured on different bridge domains or VLANs, they can coexist on the same device.
- The load balancing method required in the Layer 2 VPN is different from the Layer 3 VPN. Layer 3 VPN and Layer 2 VPN forwarding are performed independently on the device using two different types of adjacencies; therefore, the forwarding is not impacted by having a different method of load balancing for the Layer 2 VPN.
- Starting from Cisco NX-OS Release 8.2(1), all EoMPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on M3-Series I/O modules.
- Starting from Cisco NX-OS Release 8.4(1), all EoMPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on F4-Series I/O modules.

EVCs have the following configuration guidelines and limitations:

- EFPs can be created only on Layer 3 interfaces without a switchport or IP address configuration.
- EFPs are not supported on subinterfaces.
- The total number of EFPs and subinterfaces that are supported in a system is 4000.
- The following features are not supported:
  - Service instance (Ethernet flow point [EFP]) group support.
  - EVC cross-connect and connect forwarding services.
  - Ethernet service protection features such as Ethernet Operations, Administration, and Maintenance (EOAM), Connectivity Fault Management (CFM), or Ethernet Local Management Interface (E-LMI).
  - Access control lists (ACLs).

Layer 2 VPN internetworking has the following configuration guidelines and limitations:

- Ethernet interworking for a raw Ethernet port or a VLAN trunk is not supported. Traffic streams are not kept separate when traffic is sent between transport types.
- The following restrictions apply to Layer 2 VPN internetworking and VLAN:
  - There must be a one-to-one relationship between an attachment circuit and the pseudowire. Point-to-multipoint or multipoint-to-point configurations are not supported.
  - The routing protocols for point-to-point operation on the customer edge (CE) devices must be configured when configuring an Ethernet to a non-Ethernet setup.
  - The Ethernet or VLAN must contain only two IP devices: the PE device and the CE device. The PE device performs proxy Address Resolution Protocol (ARP) and responds to all ARP requests that it receives. Therefore, only one CE and one PE device should be on the Ethernet or VLAN segment.
  - When you change the interworking configuration on an Ethernet PE device, clear the ARP entry on the adjacent CE device so that it can learn the new MAC address. Otherwise, you might experience traffic drops.
- The following restriction applies if you configure Layer 2 VPN internetworking between Ethernet and VLAN with Cisco Catalyst switches as the CE devices:
  - The Spanning Tree Protocol (STP) is supported for Ethernet interworking. Ethernet interworking between an Ethernet port and a VLAN supports STP only on VLAN 1. Configure VLAN 1 as a non native VLAN.

Field Descriptions for Tunnel Interfaces

This section includes the following field descriptions for tunnel interfaces:

- Tunnel: Details Tab: Tunnel Details Section, page 27-491
- Tunnels: Details Tab: Source Section, page 27-491
- Tunnel: Statistics Tab, page 27-491

Tunnel: Details Tab: Tunnel Details Section

Table 27-1 Tunnel: Details: Tunnel

Field	Description
Device	*Display only.* Name of device where tunnel interface exists.
Tunnel ID	*Display only.* Tunnel interface number.
Description	String that describes the tunnel interface.
Admin Status	Administrative status of the tunnel interface. The default is down.
Oper Status	Operational status of the tunnel interface.
MTU	MTU value for this tunnel.
IP Address	IPv4 address in dotted decimal notation.
Net mask	Network mask for the IPv4 address, in dotted decimal notation.
IPv6 Address	IPv6 prefix in x:x:x::x/length format.

Tunnels: Details Tab: Source Section

Table 27-2 Tunnels: Details: Source

Field	Description
Local Endpoint	
Interface	Interface for the tunnel source address.
IP Address	IPv4 address, in dotted decimal notation for the tunnel source address.
Remote Endpoint	
Host Name	Device name for tunnel destination.
IP Address	IPv4 address, in dotted decimal notation for the tunnel destination address.

Tunnel: Statistics Tab

Table 27-3 Tunnel: Statistics Tab

Field	Description
Status	Status of statistics collection. Roll over Status to get a popup tip.
Select Parameters	List of statistics that can be gathered on tunnel interfaces.
Show Overview Chart	Overview popup of statistics.

Platform Support

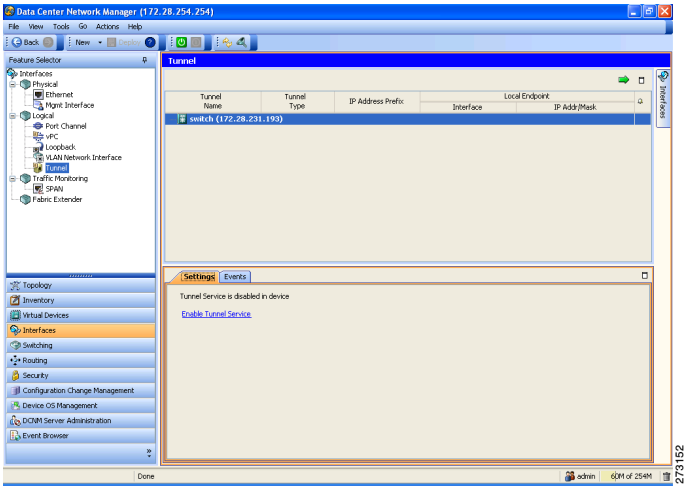
The following platform supports this feature. For platform-specific information, including guidelines and limitations, system defaults, and configuration limits, see the corresponding documentation.

Platform	Documentation
Cisco Nexus 7000 Series Switches	Cisco Nexus 7000 Series Switches Documentation

Configuring Ethernet over MPLS

You can access IP tunnels from the Interfaces feature selection. Figure 27-1 shows how to configure IP tunnels.

Figure 27-1 Configuring Tunnel Interfaces



For more information about Cisco DCNM features, see the *Fundamentals Configuration Guide, Cisco DCNM for LAN, Release 5.x* .

This section includes the following topics:

- Enabling Ethernet Virtual Circuits, page 27-492
- Configuring Ethernet Flow Points, page 27-494
- Associating an Ethernet Flow Point to a Bridge Domain, page 27-496

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Enabling Ethernet Virtual Circuits

SUMMARY STEPS

1. **configure terminal**

2. **feature evc**
3. **exit**
4. (Optional) **show feature**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**feature evc** **Example:** switch(config)# feature evc	Enables Ethernet virtual circuits on the device.
Step 3	**exit** **Example:** switch(config)# exit switch#	Exits global configuration mode.
Step 4	**show feature** **Example:** switch# show feature	(Optional) Displays the status of features on a device.
Step 5	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**feature tunnel** **Example:** switch(config)# feature tunnel	Enables tunnels on the device.
Step 3	**exit** **Example:** switch(config)# exit	Exists configuration mode.

	Command	Purpose
Step 4	**show feature** **Example:** switch# show feature	(Optional) Displays which features are enabled on the device.
Step 5	**copy running-config startup-config** **Example:** switch(config-if)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Step 1 From the Feature Selector pane, choose **Interfaces > Logical > Tunnel**.

The available devices appear in the Summary pane.

Step 2 From the Summary pane, double-click the device that you want to enable IP tunneling on.

Step 3 From the Details pane, click the **Enable Tunnel Service** link if present.

Step 4 From the menu bar, choose **File > Deploy** to apply your changes to the device.

RELATED TOPICS

- Configuring Ethernet Flow Points, page 27-494

Configuring Ethernet Flow Points

Repeat this task for each EFP that you want to configure.

BEFORE YOU BEGIN

Ensure that you have enabled EVCs.

Restrictions

- You can configure either a single default EFP or one or more EFPs with dot1q encapsulation on a parent port, but not both. Do not configure the **encapsulation default** command under an EFP unless it is the only service instance configured on the parent port.
- A maximum of 16 rewrite operations are supported per parent port on Cisco Nexus devices.
- No two EFPs for a parent port can have the same rewrite configuration.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
   or
   interface port-channel *port-channel-number*
3. **no ip address** *ip-address mask*
4. [**no**] **service instance** *service-instance-id* **ethernet**

5. (Optional) **description** *description*
6. **encapsulation** {**default** | **dot1q** *vlan-id*}
7. (Optional) **rewrite ingress tag push dot1q** *vlan-id* **symmetric**
8. (Optional) **rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric**
9. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**interface ethernet** *slot/port* or **interface port-channel** *port-channel-number* **Example:** switch(config)# interface ethernet 2/1 switch(config-if)# or switch(config)# interface port-channel 1 switch(config-if)#	Enters interface configuration mode.
Step 3	**no ip address** *ip-address mask* switch(config-if)# no ip address 10.1.1.1 255.255.255.0	Disables IP processing on an interface.
Step 4	[**no**] **service instance** *service-instance-id* **ethernet** **Example:** switch(config-if)# service instance 1 ethernet switch(config-if-srv)#	Enters interface services configuration mode and configures an EFP on the interface. • The *service-instance-id* argument is a unique per-interface identifier for this EFP. The valid range is from 1 to 4000. The range might be restricted due to resource constraints. **Note** You can use the **no** form of this command to delete the EFP and the associated configuration.
Step 5	**description** *description* **Example:** switch(config-if-srv)# description EFP1forTest	(Optional) Adds a description to this service instance configuration. The maximum range for the *description* argument is 80 alphanumeric, case-sensitive characters.

	Command	Purpose
Step 6	**encapsulation** {**default** | **dot1q** *vlan-id*} **Example:** `switch(config-if-srv)# encapsulation default` or **Example:** `switch(config-if-srv)# encapsulation dot1q 10`	Specifies that all dot1q frames that are otherwise unmatched by any other EFP are matched to this EFP. **Note** You can enter the **encapsulation default** command only once in a parent port configuration. or Configures the matching criteria for mapping dot1q frames on an ingress interface to this EFP. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967.
Step 7	**rewrite ingress tag push dot1q** *vlan-id* **symmetric** **Example**: `switch(config-if-srv)# rewrite ingress tag push dot1q 30 symmetric`	(Optional) Adds one VLAN tag to the incoming dot1q frame and symmetrically applies the operation to the ingress and egress frames. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967. **Note** This command is supported only on an EFP configured with the **encapsulation default** command.
Step 8	**rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric** **Example**: `switch(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 20 symmetric`	(Optional) Rewrites one VLAN tag in the incoming dot1q frame and symmetrically applies the operation to the ingress and egress frames. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967. **Note** This command is supported only on an EFP configured with the **encapsulation dot1q** command.
Step 9	**copy running-config startup-config** **Example:** `switch(config-if-srv)# copy running-config startup-config`	(Optional) Saves this configuration change.

Associating an Ethernet Flow Point to a Bridge Domain

BEFORE YOU BEGIN

Ensure that you have configured the EFP.

Restrictions

Switchport VLANs and EFPs cannot be associated with the same bridge domain.

SUMMARY STEPS

1. **configure terminal**
2. **system bridge-domain** *id* [**-***id* | **-***id*,...,*id*-*id*]
3. [**no**] **bridge-domain** *domain-id*
4. **member** *interface slot/port* **service instance** *service-instance-id*
5. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**system bridge-domain** *id* [**-***id* | **-***id*,...,*id*-*id*] **Example:** `switch(config)# system bridge-domain 10-50,100-500`	Identifies the IDs that are available for bridge-domain configurations. • The valid range for the *id* argument is from 2 to 967. • The optional **-***id* keyword and argument combination identifies the last ID in a range of contiguous IDs. The hyphen (-) is required. • The optional list of ID ranges are separated by commas (,). Do not type the ellipses (...).
Step 3	[**no**] **bridge-domain** *domain-id* **Example:** `switch(config)# bridge-domain 10` `switch(config-bdomain)#`	Enters bridge-domain configuration mode and configures a bridge domain. • The *domain-id* argument is a unique identifier for the bridge domain and underlying VLAN to be created. The valid range is defined by the system-bridge-domain configuration. **Note** You can use the **no** form of this command to remove the bridge-domain configuration including port associations. Removing the bridge-domain configuration does not remove the underlying VLAN. If a VLAN is associated with a bridge domain, you cannot remove the VLAN without first removing the bridge domain. To remove the underlying VLAN, use the **no vlan** command after you remove the bridge domain.

	Command	Purpose
Step 4	**member** *interface slot/port* **service instance** *service-instance-id* **Example:** `switch(config-bdomain)# member ethernet 2/1 service instance 1`	Binds a service instance to this bridge domain. • The *interface slot/port* argument identifies the interface under which the service instance is configured. • The *service-instance-id* argument identifies the service instance to be bound. The valid range is from 1 to 4000.
Step 5	**copy running-config startup-config** **Example:** `switch(config-bdomain)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying the Ethernet over MPLS Configuration

To verify EoMPLS configuration information, perform one of the following tasks:

Command	Purpose
show bridge-domain	Displays information about bridge domains that are configured on the device.
show ethernet service instance [**detail**]	Displays information about service instances that are configured on the device.
show ethernet service instance interface ethernet *slot/port* [**detail**]	Displays information about service instances that are configured on an interface.
show ethernet service instance id *service-instance-id* **interface ethernet** *slot/port* [**detail**]	Displays information about a specific service instance that is configured on an interface.
show interface description	Displays a description for interfaces.
show interface ethernet	Displays interface status and information.
show interface ethernet *slot/port* **brief**	Displays brief information about the interface.
show interface ethernet *slot/port* **counters**	Displays in and out counters for the interface.
show interface status	Displays the interface line status.
show l2vpn service all detail	Displays information about Layer 2 VPN services.
show vlan *bridge-domain-id*	Displays EFPs associated with a bridge domain.

Monitoring Tunnel Interfaces

You can configure DCNM to collect tunnel interface statistics. Choose **Interfaces > Logical > Tunnel** from the Feature Selector and navigate to the interface that you want to collect statistics on.

You see the Port Traffic Statistics window. You can collect statistics on input and output (packet and byte) counters, broadcast, multicast, and unicast traffic.

See the *Fundamentals Configuration Guide, Cisco DCNM for LAN, Release 5.x* , for more information on collecting statistics for layer 3 interfaces.

Configuration Examples for Ethernet over MPLS

This section includes the following topics:

Example: EVC

The following example shows how to configure three bridge-domain interfaces and service instances for EoMPLS:

```
system bridge-domain 10-50,100-500

bridge-domain 10
  member Ethernet 2/1 service instance 1

bridge-domain 20
  member Ethernet 2/1 service instance 2

bridge-domain 30
  member Ethernet 2/2 service instance 3

interface Ethernet 2/1
  service instance 1
    encapsulation dot1q 10

interface Ethernet 2/1
  service instance 2
    encapsulation dot1q 11
    rewrite ingress translate 1-to-1 dot1q 20 symmetric

interface Ethernet 2/2
  service instance 3
    encapsulation default
    rewrite ingress push dot1q 30 symmetric
```

Example: Xconnect

This example shows how to configure an Xconnect context for EoMPLS to connect PE devices:

PE1

```
feature-set mpls
feature ospf
feature mpls ldp
feature mpls l2vpn

l2vpn
feature evc

vlan 1,100


l2vpn xconnect context foo
```

```
    member Ethernet1/1
    member pseudowire1 20.0.0.4 200 encapsulation mpls

interface Ethernet1/1
    no shutdown

interface Ethernet1/24
    mpls ip
    ip address 1.1.1.1/24
    ip router ospf pe1 area 0.0.0.0
    no shutdown

interface loopback0
    ip address 20.0.0.1/32
    ip router ospf pe1 area 0.0.0.0

mpls ldp configuration
    discovery targeted-hello accept
    router-id Lo0 force
    neighbor 20.0.0.4 targeted
router ospf pe1
```

Hostname P1

```
feature-set mpls
hostname p1
feature ospf
feature mpls ldp
interface Ethernet1/25
    mpls ip
    ip address 1.1.1.2/24
    ip router ospf p1 area 0.0.0.0
    no shutdown
interface Ethernet1/48
    mpls ip
    ip address 2.1.1.1/24
    ip router ospf p1 area 0.0.0.0
    no shutdown
interface loopback0
    ip address 20.0.0.2/32
    ip router ospf p1 area 0.0.0.0

mpls ldp configuration
    router-id Lo0 force
router ospf p1
```

Hostname P2

```
feature-set mpls
hostname p2
feature ospf
feature mpls ldp
interface Ethernet2/1
    mpls ip
    ip address 2.1.1.2/24
    ip router ospf p2 area 0.0.0.0
    no shutdown
interface Ethernet2/24
    mpls ip
    ip address 3.1.1.1/24
    ip router ospf p2 area 0.0.0.0
    no shutdown
interface loopback0
    ip address 20.0.0.3/32
```

```
    ip router ospf p2 area 0.0.0.0
mpls ldp configuration
    router-id Lo0 force
router ospf p2
```

PE2

```
feature-set mpls

feature telnet
feature ospf
feature mpls ldp
feature mpls l2vpn

l2vpn
feature evc

l2vpn xconnect context foo
    member Ethernet2/47
    member pseudowire1 20.0.0.1 200 encapsulation mpls

interface Ethernet2/25
    mpls ip
    ip address 3.1.1.2/24
    ip router ospf pe2 area 0.0.0.0
    no shutdown

interface Ethernet2/47
    no shutdown

interface loopback0
    ip address 20.0.0.4/32
    ip router ospf pe2 area 0.0.0.0

mpls ldp configuration
    discovery targeted-hello accept
    router-id Lo0 force
    neighbor 20.0.0.1 targeted
router ospf pe2
```

Additional References

For additional information related to configuring EVCs, see the following sections:

- Related Documents, page 27-502
- MIBs, page 27-502

Related Documents

Related Topic	Document Title
Interface commands	*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*
VLAN commands	Cisco Nexus 7000 Series NX-OS Layer 2 Switching Command Reference
Any Transport over MPLS	"Configuring Any Transport over MPLS" chapter

MIBs

MIBs	MIBs Link
• BRIDGE-MIB • CISCO-EVC-MIB • CISCO-VLAN-MEMBERSHIP-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/dc-os/mibs

Feature History for Ethernet Virtual Circuits

Table 27-4 lists the release history for this feature.

Table 27-4 Feature History for Ethernet Virtual Circuits

Feature Name	Releases	Feature Information
Ethernet over MPLS (EoMPLS)	8.4(1)	Starting from Cisco NX-OS Release 8.4(1), all EoMPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on F4-Series I/O modules.
Ethernet over MPLS (EoMPLS)	8.2(1)	Starting from Cisco NX-OS Release 8.2(1), all EoMPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on M3-Series I/O modules.
Ethernet over MPLS (EoMPLS)	6.2(2)	This feature was introduced. The following commands were introduced or modified: **bridge-domain**, **encapsulation**, **inherit port-profile**, **interface pseudowire**, **l2vpn xconnect context**, **member**, **port-profile**, **rewrite ingress tag**, **show bridge-domain**, **show ethernet service instance**, **show interface**, **show interface ethernet**, **show interface pseudowire, system bridge-domain**.
IP tunnels in VDC other than default	4.2(1)	This features was introduced.

Configuring EoMPLS Layer 2 VPN Graceful Restart

This chapter describes how to configure Label Distribution Protocol (LDP) Graceful Restart (GR) for the Ethernet over MPLS (EoMPLS) Layer 2 VPN Graceful Restart feature.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About EoMPLS Layer 2 VPN Graceful Restart

This section includes the following topics:

EoMPLS Layer 2 VPN Graceful Restart

The EoMPLS L2VPN Graceful Restart feature enables a switch configured with the Label Distribution Protocol (LDP) Graceful Restart (GR) to assist its neighboring switches to recover gracefully from an interruption in service. The neighboring switches must be configured with Multiprotocol Label Switching (MPLS) LDP Stateful Switchover/Nonstop Forwarding (SSO/NSF) Support and Graceful Restart (GR). The MPLS LDP GR functions in helper mode, which means that it can only help recover switches that are enabled with MPLS SSO/NSF and GR. If the switch with the LDP GR fails, its peer switches cannot help it recover.

When you enable MPLS LDP GR on a switch that peers with an MPLS LDP SSO/NSF-enabled switch, the SSO/NSF-enabled switch can maintain its forwarding state when the LDP session between them is interrupted. While the SSO/NSF-enabled switch recovers, the peer switch forwards packets using stale information. This process enables the SSO/NSF-enabled switch to become operational more quickly.

Label Distribution Protocol Graceful Restart

Label Distribution Protocol Graceful Restart (LDP GR) works in Strict Helper mode, where it helps a neighboring switch configured with MPLS LDP Stateful Switchover/Nonstop Forwarding (SSO/NSF) to recover from disruption in service without losing its MPLS forwarding state.

In a typical scenario, three switches and LDP sessions are established between the first switch (Switch 1) and the second switch (Switch 2), as well as between Switch 2 and the third switch (Switch 3). Switch 2 is configured with MPLS LDP SSO/NSF. Switches 1 and 3 are configured with MPLS LDP GR and a label switched path (LSP) is established between Switch 1 and Switch 3.

In this scenario, if there is a disruption of service at Switch 2, Switches 1 and 3 (configured with LDP GR) help Switch 2 (configured with LDP SSO/NSF) to recover from the disruption by performing the following actions:

- Switch 1 notices an interruption in service with Switch 2. (Switch 3 also performs the same actions in this process.)
- Switch 1 marks all the label bindings from Switch 2 as stale but continues to use the bindings for MPLS forwarding.
- Switch 1 reestablishes an LDP session with Switch 2 but keeps its stale label bindings.
- Both switches readvertise their label binding information. If Switch 1 relearns a label from Switch 2 after the session has been established, the stale flags are removed.

Licensing Requirements for EoMPLS Layer 2 VPN Graceful Restart

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco DCNM	IP tunnels require a LAN Enterprise license. For a complete explanation of the DCNM licensing scheme and how to obtain and apply licenses, see the *Cisco DCNM Installation and Licensing Guide, Release 5.x.*
Cisco NX-OS	Layer 2 MVPNs require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations for EoMPLS Layer 2 VPN Graceful Restart

EoMPLS Layer 2 VPN Graceful Restart has the following configuration guidelines and limitations:

- Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) Graceful Restart (GR) is supported in Strict Helper mode.
- Tag Distribution Protocol (TDP) sessions are not supported. Only LDP sessions are supported.
- MPLS LDP GR cannot be configured on label-controlled ATM (LC-ATM) interfaces.

Configuring EoMPLS Layer 2 VPN Graceful Restart

Repeat this task to configure LDP Graceful Restart on each neighboring NX-OS device.

SUMMARY STEPS

1. **configure terminal**
2. **mpls ldp configuration**
3. **mpls ldp graceful-restart**
4. **interface** *type slot/port*
5. **mpls ip**
6. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**mpls ldp configuration** **Example:** `switch(config)# mpls ldp configuration` `switch(config-ldp)#`	Enters Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) configuration mode.
Step 3	**mpls ldp graceful-restart** **Example:** `switch(config-ldp)# mpls ldp graceful-restart`	Enables the switch to protect the LDP bindings and MPLS forwarding state during a disruption in service.

	Command	Purpose
Step 4	**interface** *type slot/port* **Example:** `switch(config-ldp)# interface ethernet 2/12` `switch(config-if)#`	Enters interface configuration mode and configures the specified interface.
Step 5	**mpls ip** **Example:** `switch(config-if)# mpls ip`	Configures MPLS hop-by-hop forwarding on this interface.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying the EoMPLS Layer 2 VPN Graceful Restart Configuration

To verify PW and VFI configuration information, perform one of the following tasks:

Command	Purpose
show mpls ldp graceful-restart	Displays Graceful Restart sessions and session parameters.
show mpls ldp neighbor graceful-restart	Displays the Graceful Restart information for the LDP sessions.
show run mpls ldp all	Displays information about all of the running LDP sessions.

Monitoring Tunnel Interfaces

You can configure DCNM to collect tunnel interface statistics. Choose **Interfaces > Logical > Tunnel** from the Feature Selector and navigate to the interface that you want to collect statistics on.

You see the Port Traffic Statistics window. You can collect statistics on input and output (packet and byte) counters, broadcast, multicast, and unicast traffic.

See the *Fundamentals Configuration Guide, Cisco DCNM for LAN, Release 5.x* , for more information on collecting statistics for layer 3 interfaces.

Configuration Examples for EoMPLS Layer 2 VPN Graceful Restart

The following example shows how to enable MPLS LDP graceful restart:

```
feature mpls ldp
interface Ethernet2/12
    mpls ip
mpls ldp configuration
    no shutdown
    no mpls ip default-route
    backoff 15 120
    logging neighbor-changes
    logging password rollover
    logging password configuration
    no explicit-null
    graceful-restart
    graceful-restart timers forwarding-holding 120
    graceful-restart timers max-recovery 120
    graceful-restart timers neighbor-liveness 120
    holdtime 180
    discovery hello interval 5
    discovery hello holdtime 15
    discovery targeted-hello interval 10
    discovery targeted-hello holdtime 90
    no discovery targeted-hello accept
    router-id Lo0 force
    advertise-labels
    label allocate global host-routes
```

Additional References for EoMPLS Layer 2 VPN Graceful Restart

For additional information related to EoMPLS Layer 2 VPN GR, see the following sections:

- Related Documents, page 28-6
- MIBs <Optional: remove if not applicable>, page 28-6

Related Documents

Related Topic	Document Title
MPLS commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
MPLS Label Distribution Protocol	"MPLS LDP Graceful Restart" chapter

MIBs <Optional: remove if not applicable>

MIBs	MIBs Link
MPLS-LDP-STD-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/dc-os/mibs

Feature History for EoMPLS Layer 2 VPN Graceful Restart

Table 28-1 lists the release history for this feature.

Table 28-1 Feature History for EoMPLS Layer 2 VPN Graceful Restart

Feature Name	Releases	Feature Information
EoMPLS Layer 2 VPN Graceful Restart	6.2(2)	The EoMPLS Layer 2VPN Graceful Restart feature enables a switch configured with the Label Distribution Protocol (LDP) Graceful Restart (GR) to assist its neighboring switches to recover gracefully from an interruption in service.
IP tunnels in VDC other than default	4.2(1)	This features was introduced.

Configuring Virtual Private LAN Service

Note The Cisco NX-OS release that is running on a managed device may not support all the features or settings described in this chapter. For the latest feature information and caveats, see the documentation and release notes for your platform and software release.

This chapter describes how to configure Virtual Private LAN Service (VPLS) using the Cisco Data Center Network Manager (DCNM) Access Circuits (ACs) for Layer 2 Virtual Private Networks (L2VPNs) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About Virtual Private LAN Service

This section includes the following topics:

Layer 2 Services

A Layer 2 Virtual Private Network (L2VPN) enables service providers to carry multiple network services over a single converged network using Multiprotocol Label Switching (MPLS). MPLS L2VPN extends the Layer 2 domains in data centers. MPLS can be used to connect branch offices to back up data centers and also to interconnect multiple data centers in the same organization.

L2VPN services using the MPLS/IP core can be divided into two categories: wire and LAN services. The Virtual Private Wire Service (VPWS) provides point-to-point service between two customer edge (CE) devices over the provider core. The Virtual Private LAN Service (VPLS) provides point-to-multipoint service between multiple customer sites using a mesh of point-to-point pseudowires over the provider core to emulate a LAN between the sites.

Attachment Circuits

A Layer 2 circuit that connects a customer edge (CE) node to a provider edge (PE) node is known as an attachment circuit or AC. A Layer 2 VPN (L2VPN) supports only Ethernet ACs on Cisco NX-OS devices.

To cross the network core, the Layer 2 traffic is tunneled inside a pseudowire. A pseudowire is typically a Multiprotocol Label Switching (MPLS) label-switched path (LSP), or a Layer 2 Tunneling Protocol (L2TP) tunnel, or the psuedowire can be locally switched from another AC. Layer 2 VPN connects different types of circuits (that is, different types of Layer 2 ACs and pseudowires) together in different ways to implement different types of end-to-end services.

The following types of ACs are supported:

- Ethernet port mode—This AC includes all frames that are sent and received on a physical Ethernet port.
- Ethernet 802.1Q—This AC includes all frames that are sent and received with a particular VLAN tag.

- Ethernet 802.1ad (Q-in-Q)—This AC includes all frames that are sent and received with a specific outer VLAN tag and a specific inner VLAN tag. VLAN-in-VLAN (Q-in-Q) is supported only in the service instance configuration and not in the subinterface configuration.
- Ethernet QinAny—This AC includes all frames that are sent and received with a specific outer VLAN tag and any inner VLAN tags, as long as the inner VLAN tag is not used on another subinterface.

An attachment circuit can participate in a Virtual Private LAN Service (VPLS) through a bridge domain. The Layer 2 switch port interfaces can also participate in VPLS forwarding. You can configure link bundles (port channels) with Ethernet Virtual Circuits (EVCs) to provide encapsulation types for link bundles.

Pseudowire Interface

A pseudowire (PW) is a mechanism for emulating various networking or telecommunications services across packet-switched networks that use Ethernet, IP, or MPLS. A pseudowire interface (also known as a PW) in Cisco NX-OS is a logical interface type that represents a PW so that it can be consistently characterized in all communication and operations throughout the system.

You can create a static PW or dynamic PW configuration in pseudowire interface mode. Long form pseudowire interfaces must be explicitly configured using the appropriate Cisco NX-OS commands. Short-term, also known as auto-generated or dynamic, PWs are programmatically created and destroyed; you cannot configure a short-term PW. PW configurations can also be imported using a port profile.

With VPLS, different sites can share an Ethernet broadcast domain via PWs, providing any-to-any connectivity. VPLS uses a full mesh of Ethernet PWs to emulate a LAN segment or broadcast domain that is capable of learning and forwarding, based on Ethernet MAC addresses. The PW if-index is used as a handle for identification throughout the system; MAC entries are also acquired against these PWs.

Control Word

According to RFC 4448, if a pseudowire (PW) is sensitive to packet misordering and is being carried over an MPLS packet switched network (PSN) that uses the contents of the MPLS payload to select the Equal Cost Multipath (ECMP), the PW must employ a mechanism that prevents packet misordering. This is necessary because ECMP implementations may examine the first nibble after the MPLS label stack to determine whether the labeled packet is IP or not. If the source MAC address of an Ethernet frame carried over the pseudowire without a control word present begins with 0x4 or 0x6, it can be mistaken for an IPv4 or an IPv6 packet. Depending on the configuration and topology of the MPLS network, this can lead to a situation where all packets for a given PW do not follow the same path, increasing out-of-order frames on a given PW or causing Operations, Administration, and Maintenance (OAM) packets to follow a different path than the actual traffic.

The Control Word Support feature provides the ability to sequence individual frames on the pseudowire, avoid ECMP paths, and perform OAM mechanisms including Virtual Circuit Connectivity Verification (VCCV).

Virtual Forwarding Interface

A virtual forwarding interface (VFI) defines the configuration and the membership of the core pseudowires in the VPLS. A VFI is a virtual Layer 2 bridge that connects attachment circuits (physical Ethernet ports, logical Ethernet ports, or PWs) from customer edge (CE) devices to virtual circuits (VCs). The VFI is allocated an interface type and index in the system and is used by L2VPN and other components as an identifier.

Bridge Domain

A bridge domain is a generic object that represents a Layer 2 broadcast domain on a device. VPLS uses a bridge domain to define a point-to-multipoint layer 2 service.

Creating a bridge domain also creates the underlying VLAN, if it does not already exist. There is a one-to-one mapping of bridge domains to VLANs; bridge domain 100 maps to VLAN 100.

Ethernet Virtual Circuits

An Ethernet Virtual Circuit (EVC) as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint Layer 2 circuit. It is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer.

Ethernet Flow Point

An Ethernet Flow Point (EFP) is the instantiation of an EVC on a specific interface on a device. The EFP interface representation is similar to that of a subinterface that maintains the parent-child relationship with the port.

The EFP interface is a Layer 2 logical interface. Any Layer 2 feature, protocol, or application that functions on a switchport is equally applicable to an EFP, all though some constraints might apply. Similar to a physical port, the interface state machine and forwarding behavior for the EFP depends on the service to which it belongs.

An EFP interface, also known as a service instance, is implicitly created when you configure an Ethernet service instance on a port. An EFP can be configured under a physical or logical parent port. Each service instance has its own configuration submode. Different features that apply to the service instance can be configured in that submode.

Because a single parent port can support multiple service instances, several EFPs can be associated with the port, with each EFP as part of a different EVC. For this reason, whenever a service instance is configured on a port, the port is internally brought up in trunk mode.

Note The EVC represents a bridge domain. An EFP is an instance of an Ethernet flow on a particular interface, that belongs to a bridge domain. The Ethernet flow, not the entire port, belongs to the bridge domain.

Flow per EFP

EVCs can identify flows based on multiple criteria in the Layer 2 header. In Cisco NX-OS, the flow identification for devices with Enhanced Address Recognition Logic (Earl) 8 hardware is based on matching the VLAN tag of the incoming packet. If the incoming packet has multiple VLAN tags only, the outer tag is used for traffic mapping to EFP.

Encapsulation defines the matching criteria that maps a VLAN to the service instance. A single VLAN ID can be configured for an exact match of the outermost tag. Any VLAN ID that is not specifically configured on an EFP or subinterface is treated as if it is implicitly configured for default encapsulation. On a parent port, you can configure either a single default EFP or one or more EFPs with explicit encapsulation, but not both.

Border Gateway Protocol Auto Discovery

Border Gateway Protocol Auto Discovery (BGP-AD) automatically detects when provider edge (PE) devices are added to or removed from the VPLS domain, eliminating the need to manually configure PWs. BGP-AD can use either BGP or Label Distribution Protocol (LDP) signaling to exchange label binding information for supporting forwarding in an MPLS network.

The BGP-based auto discovery mechanism distributes Layer 2 VPN (L2VPN) endpoint provisioning information. BGP uses a separate L2VPN Routing Information Base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 VFI is configured. Prefix and path information is stored in the L2VPN database, allowing BGP to make best-path decisions. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to set up a pseudowire mesh to support L2VPN-based services.

The VPLS BGP Signaling feature enables you to use BGP as the control plane protocol for both auto discovery and signaling in accordance with RFC 4761. Internal BGP (iBGP) peers exchange L2VPN AFI/SAFI update messages with L2VPN information to perform both auto discovery and signaling. The BGP multiprotocol Network Layer Reachability Information (NLRI) consists of a Route Distinguisher (RD), VPLS Endpoint ID (VE ID), VE Block Offset (VBO), VE Block Size (VBS), and Label Base (LB). Each NLRI consists of block labels such as LB, LB+1,...., LB+VBS-1. The NLRI is exchanged between BGP devices for BGP auto discovery with BGP signaling.

Label Distribution Protocol (LDP)–based signaling follows the procedures specified in RFC 4447, which states that one provider edge device (PE1) sends a Label Mapping message to another PE device (PE2) to establish an LDP session in one direction. If the message is processed successfully, and there is no LDP session for the pseudowire in the opposite (PE2–to–PE1) direction, then PE2 sends a Label Mapping message to PE1.

For PE1 to begin signaling PE2, PE1 must know the address of the remote PE2. This information can be configured at PE1, or it can be generated dynamically through an auto–discovery procedure. The egress PE (PE1), which has knowledge of the ingress PE, initiates the setup by sending a Label Mapping message to the ingress PE (PE2), the Label Mapping message contains the FEC Tag Limit Values (TLV).

When the PE2 receives a Label Mapping message, PE2 interprets the message as a request to set up a pseudowire whose endpoint, PE2 is the forwarder. A Virtual Circuit (VC) or a pseudowire label is used to process packets at each PE device. Each PE device must reserve a PW label (local label) and advertise it to the peer. The VC label bindings exchanged over the targeted LDP session use the Forwarding Equivalence Classes (FEC) element type 128 via the LDP downstream unsolicited mode. Only one targeted session is created for multiple VCs between the PEs. If there already is a targeted session between the PEs by another application, then that session will be used. LDP will use the FEC type 128 to determine that the message is for the AToM application. LDP FEC 129 is used with auto discovery.

Note VPLS with LDP signaling and no auto discovery is the most widely deployed solution.

MAC Address Support

Layer 2 VPN (L2VPN) MAC address support is enabled by default when you configure a VPLS.

MAC Address Flooding

One of the attributes of an Ethernet service is that frames sent to broadcast addresses and to unknown destination MAC addresses are flooded to all ports. To achieve flooding within the service provider network, all unknown unicast, broadcast and multicast frames are flooded over the corresponding pseudowires (PWs) to all provider edge (PE) nodes participating in the VPLS, as well as to all attachment circuits (ACs).

Multicast frames are different and do not necessarily have to be sent to all VPN members. For simplicity, the default approach of broadcasting multicast frames is used. To forward a frame, a PE must be able to associate a destination MAC address with a PW. VPLS-capable PEs have the capability to dynamically learn MAC addresses on both ACs and PWs and to forward and replicate packets across both ACs and PWs.

The MAC address table contains a list of the known MAC addresses and their forwarding information. In a typical VPLS architecture, the MAC address table and its management are distributed, which means that a copy of the MAC address table is maintained on the route processor (RP) card and the line cards.

MAC Address Forwarding

A MAC address table contains address information that the switch uses to forward traffic between ports. All MAC addresses in the address table are associated with one or more ports. The table also contains a list of all known MAC addresses and their forwarding information. To forward a frame, a provider edge (PE) device must associate a destination MAC address with a pseudowire or an attachment circuit. This type of association is provided through a static configuration on each PE device or through dynamic learning that is flooded on all bridge ports.

When Layer 2 frames are received, VPLS does a lookup of the destination MAC address to learn the source MAC address. If the destination MAC address is not present in the MAC address table, the Layer 2 frames are flooded on the VLAN on which these frames were received. Flooded frames are sent on all configured pseudowires.

When Layer 2 frames are received on a pseudowire, the source MAC address is learned from the MAC address table by using the pseudowire port identifier (peer_id). If the destination MAC address is not present in the MAC address table, the frames are flooded on Layer 2 ports. If the destination MAC 2 address is present in the MAC address table, the frames are forwarded to the Layer 2 port or to the destination peer.

MAC Address Learning

When a Layer 2 frame arrives on a bridge port, such as a pseudowire or an attachment circuit, and the source MAC address is unknown to the receiving provider edge (PE) device, the source MAC address is associated with the pseudowire or the attachment circuit. Outbound frames to the MAC address are forwarded to the appropriate pseudowire or attachment circuit.

MAC address learning uses the MAC address information that is learned from the hardware forwarding path. The number of learned MAC addresses is limited through configurable per-port and per-bridge domain MAC address limits.

MAC Address Learning Aging

A timer is associated with the MAC addresses available in the MAC table. When this timer expires, the MAC addresses become invalid and are removed from the table. The relevant MAC entries are repopulated. This event is called MAC address aging. Provider edge (PE) devices must learn remote MAC addresses and directly attached MAC addresses on customer facing ports. MAC address learning derives topology and forwarding information from packets that originate at customer sites.

MAC Address Withdrawal

VPLS MAC address withdrawal provides faster convergence by removing (or unlearning) MAC addresses that have been dynamically learned. No configuration is needed for enabling MAC address withdrawal support. Provider edge (PE) devices learn the remote MAC addresses and directly attached MAC addresses on customer-facing ports by deriving the topology and forwarding information from packets that originate at customer sites.

Layer 2 VPN Stateful High Availability

The L2VPN Stateful High Availability (HA) feature uses two supervisor modules to provide uninterrupted service during a system failure. This implementation is the same for both Ethernet over Multiprotocol Label Switching (EoMPLS) and Virtual Private LAN Service (VPLS). During a failure, when an active supervisor is down, the standby supervisor seamlessly takes over all operations without disruptions. The supervisor modules also use Nonstop Forwarding (NSF), Stateful Switchover (SSO), and Graceful Restart (GR) for Any Transport over MPLS (AToM) to recover from an interruption in the service.

Peer Label Switch Routers (LSRs) exchange label binding information in an Multiprotocol Label Switching (MPLS) network to support the forwarding process. The MPLS Label Distribution Protocol Graceful Restart feature provides a mechanism by which the forwarding state between LSRs can be maintained during interruptions such as SSO failover events and temporary loss of Label Distribution Protocol (LDP) communication between the LSRs to enable NSF for MPLS traffic.

To enable NSF for Any Transport over MPLS (AToM) traffic, the provider edge (PE) devices and the LDP peers involved in the SSO event must support GR. There is no specific configuration required for Layer 2 VPN stateful HA.

LinkSec

The LinkSec feature provides security for data centers over pseudowires using point-to-point encryption. LinkSec supports IEEE 802.1AE link-layer cryptography that provides hop-by-hop security of data in the MAC layer. Link-layer cryptography helps to ensure end-to-end data privacy while enabling the insertion of security service devices along the encrypted path.

Hop-by-Hop Encryption

In this type of deployment, data is encrypted on the egress interface of the device and decrypted on the ingress interface of the device. Data is encrypted while being transmitted on interfaces but decrypted inside devices. However, if LinkSec is unavailable on certain segments of the network, data is sent in decrypted state on these segments. The advantage of this type of deployment is that Layer 2 Virtual Private Network (L2VPN) or Multiprotocol Label Switching (MPLS) is not aware of the encryption.

Hop-by-hop encryption is the default mode of encryption in LinkSec.

Encryption and Decryption at Customer Edge Devices

After Layer 2 Virtual Private Network (L2VPN) or Multiprotocol Label Switching (MPLS) has added its label information to the frame, LinkSec encrypts both the data packet and the VLAN tag. The VLAN tag is lost and LinkSec sends the entire package across the network as a payload. In this type of deployment, data is encrypted and decrypted at customer edge (CE) devices only.

To enable this deployment, you should configure the provider edge (PE) ports in the port mode of the L2VPN operation because the VLAN tag is lost during LinkSec encryption.

This method can also be deployed by configuring the PE ports as access switch ports and mapping the packets that enter the ingress PE1 interface to an access VLAN. The packets are then forwarded using Virtual Private Lan Service (VPLS) or Ethernet over Multiprotocol Label Switching (EoMPLS) if the egress PE1 interface is configured to be part of a bridge domain of the VLAN.

MPLS Quality of Service

To maintain the quality of service (QoS) when a packet traverses both Layer 2 and Layer 3 domains, the type of service (ToS) and CoS values must be mapped to each other. CoS refers to three bits in either an Inter-Switch Link (ISL) header or an 802.1Q header that are used to indicate the priority of an Ethernet frame as it passes through a switched network.

The 802.1Q provides QoS-based matching and marking to VLAN user priority bits to provide QoS on the Gigabit Ethernet WAN interface for 802.1Q packets. Packet marking helps identify packet flows. Packet marking enables the partitioning of a network into multiple priority levels or CoS. During network congestion, packets that are marked as priority are offered a higher priority than other packets.

802.1Q input packets are classified at eight different QoS levels (0 to 7) based on the VLAN user priority bits. For 802.1Q output packets, QoS marking is done at the VLAN header to modify VLAN user priority bits. QoS services use these priority bit settings to gain traffic priority during network congestion.

Experimental Bits

EXP is a 3-bit field and part of a Multiprotocol Label Switching (MPLS) header. Experimental (EXP) bits in an MPLS header carry the priority of packets. Each label switching device along the network path honors the packet priority by queuing packets in the proper queue and servicing packets according to the priority. EXP bits define the quality of service (QoS) treatment (per-hop behavior) that a node should give to a packet. It is the equivalent of the differentiated service code point (DSCP) in the IP network. A DSCP defines a class and drop precedence. The EXP bits generally carry all information encoded in IP DSCP. However, in some cases, the EXP bits are used exclusively to encode the dropping precedence.

QoS on a Layer 2 VPN (L2VPN) network usually has two parts, an attachment circuit (AC) side and a pseudowire side. Layer 2 QoS is applied on the AC side and Layer 3 MPLS or IP QoS is applied on the pseudowire side.

Virtual Private LAN Service (VPLS) QoS is similar to Ethernet over MPLS (EoMPLS) QoS, except that QoS in VPLS is applied at ACs that participate in a VPLS bridge domain.

The core-facing MPLS interface must support a QoS policy. This QoS policy is applied on Ethernet Virtual Circuits (EVCs) and switchport interfaces. If a switchport interface participates in QoS handling, the matching criteria must include the VLAN on which VPLS forwarding is configured.

Setting the EXP bit value helps service providers who do not want to modify the value of the IP precedence field within the IP packets that are transported through their networks. By choosing different values for the Multiprotocol Label Switching (MPLS) EXP bit field, you can specify the priority that a packet requires during periods of network congestion. By default, the IP precedence value is copied into the MPLS EXP field during imposition. On the imposition path, packets are received from the AC and are sent toward the MPLS core. You can specify the MPLS EXP bits with an MPLS quality of service (QoS) policy.

By default, EXP is derived from COS for VPLS and VLAN-based EoMPLS. For port-based EoMPLS, by default, EXP is derived from the DSCP value.

Licensing Requirements for Virtual Private LAN Service

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Layer 2 MVPN requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations for Virtual Private LAN Service

Virtual Private LAN Service (VPLS) has the following configuration guidelines and limitations:

- Fabric Extender (FEX) ports are not supported as members of either XConnect or virtual forwarding instance (VFI) contexts.
- EoMPLS and VPLS can coexist on the same device.
- Ethernet over MPLS (EoMPLS) and VPLS can coexist with MPLS Layer 3 VPNs on the same device.
- VPLS and Cisco Overlay Transport Virtualization (OTV) can coexist in the same device if they are configured on different bridge domains or VLANs. A typical use case for this type of interaction involves a scenario where one cloud of the network uses OTV and the other cloud functions on an MPLS network using VPLS. A gateway facilitates data and packet forwarding between the two clouds. The OTV cloud and the MPLS cloud can be on the same physical network.
- The load balancing method required in the Layer 2 VPN is different from the Layer 3 VPN. Layer 3 VPN and Layer 2 VPN forwarding is performed independently on the device using two different types of adjacencies; therefore, the forwarding is not impacted by having a different method of load balancing for the Layer 2 VPN.
- Starting from Cisco NX-OS Release 8.2(1), all VPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on M3-Series I/O modules.
- Starting from Cisco NX-OS Release 8.4(1), all VPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on F4-Series I/O modules.

Ethernet virtual circuits (EVCs) have the following configuration guidelines and limitations:

- Ethernet flow points (EFPs) can be created only on Layer 3 interfaces without a switchport or IP address configuration.
- EFPs are not supported on subinterfaces.
- The total number of EFPs and subinterfaces that are supported in a system is 4000.
- The following features are not supported:
  - Service instance (EFP) group support.
  - EVC cross-connect and connect forwarding services.
  - Ethernet service protection features such as Ethernet Operations, Administration, and Maintenance (EOAM), Connectivity Fault Management (CFM), or Ethernet Local Management Interface (E-LMI).
  - Access control lists (ACLs).

Pseudowires have the following configuration guidelines and limitations:

- The maximum transmission unit (MTU) value of all pseudowires in a service must be the same. A pseudowire with an MTU value that differs from the MTU value of its peers will remain in a down state.
- Multicast and broadcast counters are not supported for pseudowires. All packets and bytes will be counted as unicast.

BGP-based auto discovery has the following configuration guidelines and limitations:

- BGP-based Virtual Private LAN Service (VPLS) auto discovery supports only IPv4 addresses.
- Auto discovery uses Forwarding Equivalence Class (FEC) 129 to convey endpoint information; manually configured pseudowires use FEC 128.
- Auto discovery is not supported with interautonomous system configurations.

Field Descriptions for Tunnel Interfaces

This section includes the following field descriptions for tunnel interfaces:

- Tunnel: Details Tab: Tunnel Details Section, page 29-10
- Tunnels: Details Tab: Source Section, page 29-11
- Tunnel: Statistics Tab, page 29-11

Tunnel: Details Tab: Tunnel Details Section

Table 29-1 Tunnel: Details: Tunnel

Field	Description
Device	*Display only.* Name of device where tunnel interface exists.
Tunnel ID	*Display only.* Tunnel interface number.
Description	String that describes the tunnel interface.
Admin Status	Administrative status of the tunnel interface. The default is down.

Table 29-1 Tunnel: Details: Tunnel

Field	Description
Oper Status	Operational status of the tunnel interface.
MTU	MTU value for this tunnel.
IP Address	IPv4 address in dotted decimal notation.
Net mask	Network mask for the IPv4 address, in dotted decimal notation.
IPv6 Address	IPv6 prefix in x:x:x::x/length format.

Tunnels: Details Tab: Source Section

Table 29-2 Tunnels: Details: Source

Field	Description
Local Endpoint	
Interface	Interface for the tunnel source address.
IP Address	IPv4 address, in dotted decimal notation for the tunnel source address.
Remote Endpoint	
Host Name	Device name for tunnel destination.
IP Address	IPv4 address, in dotted decimal notation for the tunnel destination address.

Tunnel: Statistics Tab

Table 29-3 Tunnel: Statistics Tab

Field	Description
Status	Status of statistics collection. Roll over Status to get a popup tip.
Select Parameters	List of statistics that can be gathered on tunnel interfaces.
Show Overview Chart	Overview popup of statistics.

Platform Support

The following platform supports this feature. For platform-specific information, including guidelines and limitations, system defaults, and configuration limits, see the corresponding documentation.

Platform	Documentation
Cisco Nexus 7000 Series Switches	Cisco Nexus 7000 Series Switches Documentation

Configuring Access Circuits for Virtual Private LAN Service

This section contains the following topics:

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Configuring an Ethernet Virtual Circuit for an 802.1Q Access Circuit

Repeat this task for each Ethernet Virtual Circuit (EVC) and Ethernet Flow Point (EFP) that you want to configure.

Restrictions

- You can configure either a single default EFP or one or more EFPs with dot1q encapsulation on a parent port, but not both. Do not configure the **encapsulation default** command under an EFP unless it is the only service instance configured on the parent port.
- A maximum of 16 rewrite operations are supported per parent port on Cisco Nexus devices.
- No two EFPs for a parent port can have the same rewrite configuration.

SUMMARY STEPS

1. **configure terminal**
2. **feature evc**
3. **interface ethernet** *slot/port*
   or
   interface port-channel *port-channel-number*
4. **no ip address** *ip-address mask*
5. [**no**] **service instance** *service-instance-id* **ethernet**
6. (Optional) **description** description
7. **encapsulation** {**default** | **dot1q** *vlan-id*}
8. (Optional) **rewrite ingress tag push dot1q** *vlan-id* **symmetric**
9. (Optional) **rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric**
10. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature evc** **Example:** `switch(config)# feature evc`	Enables Ethernet virtual circuits on the device.
Step 3	**interface ethernet** *slot/port* or **interface port-channel** *port-channel-number* **Example:** `switch(config)# interface ethernet 2/1` `switch(config-if)#` `or` `switch(config)# interface port-channel 1` `switch(config-if)#`	Enters interface configuration mode and configures an interface.
Step 4	**no ip address** *ip-address mask* **Example:** `switch(config-if)# no ip address 10.1.1.1 255.255.255.0`	Disables IP processing on an interface.
Step 5	[**no**] **service instance** *service-instance-id* **ethernet** **Example:** `switch(config-if)# service instance 1 ethernet` `switch(config-if-srv)#`	Enters interface services configuration mode and configures an EFP on the interface. • The *service-instance-id* argument is a unique per-interface identifier for this EFP. The valid range is from 1 to 4000. The range might be restricted due to resource constraints. **Note** You can use the **no** form of this command to delete the EFP and the associated configuration.
Step 6	**description** *description* **Example:** `switch(config-if-srv)# description EFP1forVPLS`	(Optional) Adds a description to this service instance configuration. • The maximum range for the *description* argument is 80 alphanumeric, case-sensitive characters.

	Command	Purpose
Step 7	**encapsulation** {**default** | **dot1q** *vlan-id*} **Example:** `switch(config-if-srv)# encapsulation default` or switch(config-if-srv)# encapsulation dot1q 10	Specifies that all dot1q frames that are otherwise unmatched by any other EFP are matched to this EFP. **Note** You can enter the **encapsulation default** command only once in a parent port configuration. or Configures the matching criteria for mapping dot1q frames on an ingress interface to this EFP. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967.
Step 8	**rewrite ingress tag push dot1q** *vlan-id* **symmetric** **Example:** `switch(config-if-srv)# rewrite ingress tag push dot1q 30 symmetric`	(Optional) Adds one VLAN tag to the incoming dot1q frame and symmetrically applies the operation to the ingress and egress frames. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967. **Note** This command is supported only on an EFP configured with the **encapsulation default** command.
Step 9	**rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric** **Example:** `switch(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 20 symmetric`	(Optional) Rewrites one VLAN tag in the incoming dot1q frame and symmetrically applies the operation to the ingress and egress frames. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967. **Note** This command is supported only on an EFP configured with the **encapsulation dot1q** command.
Step 10	**copy running-config startup-config** **Example:** `switch(config-if-srv)# copy running-config startup-config`	(Optional) Saves this configuration change.

What to Do Next

To bind this interface to a bridge domain, see the "Configuring Virtual Private LAN Service with a Bridge Domain" section.

Manually Configuring a Pseudowire Interface

You can manually configure PWs for Access Circuits (ACs) or you can use BGP auto discovery (BGP-AD) to automatically generate PWs for the VPLS domain. To configure BGP-AD, see the "Configuring a Virtual Forwarding Interface for Auto Discovery" section.

RESTRICTIONS

- If you manually configure multiple pseudowires and target different IP addresses on the same PE device for each pseudowire, do not use the same virtual circuit identifier (VC ID) to identify the pseudowires terminated at the same PE router.
- You cannot configure a pseudowire by manually configuring a neighbor on one PE device and using auto discovery on the other PE device to configure the same pseudowire in the other direction.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **interface pseudowire** *pw-id*
3. (Optional) **control word** {**exclude** | **include**}
4. (Optional) **description**
5. **mtu** *size*
6. **neighbor** *peer-ip-address vc-id*
7. **encapsulation mpls**
8. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **interface pseudowire** *pw-id* **Example:** `switch(config)# interface pseudowire 12` `switch(config-if-pseudowire)#`	Enters interface pseudowire configuration mode and configures a static pseudowire logical interface. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. **Note** You can use the **no** form of this command to delete the pseudowire interface and the associated configuration.

	Command	Purpose
Step 3	**control-word** {**exclude** | **include**} **Example:** `switch(config-if-pseudowire)# control-word include`	(Optional) Enables control-word support. • The **include** or **exclude** keywords specify whether the control word will or will not be included in the pseudowire packet. • If you do not enable control word support in the pseudowire configuration, the default is autosense. **Note** A device can receive a packet with or without the control word and the control word capability is negotiated with the peer. However, the device will not be able to generate a sequence number in the control word if the control word is added to the ingress device.
Step 4	**description** *description* **Example:** `switch(config-if-pseudowire)# description longform`	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric, case-sensitive characters.
Step 5	**mtu** *size* **Example:** `switch(config-if-pseudowire)# mtu 1400`	(Optional) Configures the maximum transmission unit (MTU) size, in bytes, for this interface. • The valid range for the *size* argument is 576 to 9216. The default is 1500.
Step 6	**neighbor** *peer-ip-address vc-id* **Example:** `switch(config-if-pseudowire)# neighbor 10.2.2.2 100`	Configures an emulated virtual circuit for this interface. • The combination of the *peer-ip-address* and *vc-id* arguments must be unique on a device. • The peer IP address is the address of the provider edge (PE) peer. • The *vc-id* argument is an identifier for the virtual circuit between devices. The valid range is from 1 to 4294967295.
Step 7	**encapsulation mpls** **Example:** `switch(config-if-pseudowire)# encapsulation mpls` `switch(config-pseudowire-mpls)#`	Enters pseudowire MPLS configuration mode and specifies MPLS encapsulation for this interface.
Step 8	**copy running-config startup-config** **Example:** `switch(config-pseudowire-mpls)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring a Virtual Forwarding Interface for Static Pseudowires

BEFORE YOU BEGIN

Ensure that you have configured the PWs.

RESTRICTIONS

- You can configure both auto discovered and manually configured pseudowires in a single virtual forwarding instance (VFI). However, the pseudowires cannot go to the same peer PE device.
- You cannot configure a pseudowire by manually configuring a neighbor on one PE device and using auto discovery on the other PE device to configure the same pseudowire in the other direction.

SUMMARY STEPS

1. **c**onfigure terminal
2. [**no**] **l2vpn vfi context** *vfi-name*
3. (Optional) **description** *description*
4. **vpn** *vpn-id*
5. **member pseudowire** *pw-id*
6. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **l2vpn vfi context** *vfi-name* **Example:** `switch(config)# l2vpn vfi context foo` `switch(config-l2vpn-vfi)#`	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) between two or more separate networks. • The *vfi-name* argument is a unique per-interface identifier for this VFI. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the VFI and the associated configuration.
Step 3	**description** *description* **Example:** `switch(config-l2vpn-vfi)# description PWsforVPLS`	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric characters.

	Command	Purpose
Step 4	**vpn** *vpn-id* **Example:** `switch(config-l2vpn-vfi)# vpn 100`	Configures a Virtual Private Network (VPN) ID on a VFI context. • The valid range is from 1 to 4294967295.
Step 5	**member pseudowire** *pw-id* **Example:** `switch(config-l2vpn-vfi)# member pseudowire 12`	Binds a static pseudowire to this VFI. • This command is supported for a static pseudowire only. • The *pw-id* argument is a unique per-interface identifier for a static pseudowire. The valid range is from 1 to 8192. • Repeat this step for each static pseudowire to be associated with this VFI.
Step 6	**copy running-config startup-config** **Example:** `switch(config-l2vpn-vfi)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring a Virtual Forwarding Interface for Auto Discovery

Perform just one of the following tasks:

- Configuring BGP Auto Discovery and BGP Signaling, page 29-18
- Configuring BGP Auto Discovery and LDP Signaling, page 29-22

Configuring BGP Auto Discovery and BGP Signaling

RESTRICTIONS

- You can configure both auto discovered and manually configured pseudowires in a single virtual forwarding instance (VFI). However, the pseudowires cannot go to the same peer PE device.
- You cannot configure a pseudowire by manually configuring a neighbor on one PE device and using auto discovery on the other PE device to configure the same pseudowire in the other direction.
- After enabling VPLS autodiscovery, if you manually configure a neighbor by using the member command and both peers are in autodiscovery mode, each peer will receive discovery data for that VPLS. To prevent peers from receiving data for the VPLS domain, manually configure route target (RT) values. For information, see the "Customizing BGP-Based Auto Discovery Settings (optional)" section.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **l2vpn vfi context** *vfi-name*
3. (Optional) **description** *description*
4. **vpn** *vpn-id*

5. **autodiscovery bgp signaling bgp**
6. **ve id** *ve-id-number*
7. **ve range** *range*
8. **router bgp** *as-number*
9. **bgp graceful restart**
10. **neighbor** *peer-ip-address vc-id* **remote as** *as-number*
11. **address-family l2vpn vpls**
12. **neighbor** [*peer-ip-address* | *peer-group-name*] **activate**
13. **neighbor** [*peer-ip-address* | *peer-group-name*] **send-community extend**
14. **neighbor** [*peer-ip-address* | *peer-group-name*] **suppress-signaling-protocol ldp**
15. Repeat Steps 11 to 15 to configure additional neighbors in an L2VPN address family.
16. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **l2vpn vfi context** *vfi-name* **Example:** `switch(config)# l2vpn vfi context foo` `switch(config-l2vpn-vfi)#`	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) context between two or more separate networks. • The *vfi-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 3	**description** *description* **Example:** `switch(config-l2vpn-vfi)# description PWsforVPLS`	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric characters.
Step 4	**vpn** *vpn-id* **Example:** `switch(config-l2vpn-vfi)# mtu 1400`	Configures a Virtual Private Network (VPN) ID on a VFI context. • The valid range is from 1 to 4294967295.

	Command	Purpose
Step 5	**autodiscovery bgp signaling bgp** **Example:** `switch(config-l2vpn-vfi)# autodiscovery bgp signaling bgp`	Enables BGP auto discovery and BGP signaling.
Step 6	**ve id** *ve-id-number* **Example:** `switch(config-l2vpn-vfi)# ve id 1`	Configures a VPLS Endpoint ID (VEID) for the NLRI exchanged between BGP devices. • Repeat this step to add each additional VE ID. The VE ID must be unique within the same VPLS domain for all PE devices. **Note** Numbering sequences such as 1,2,3 or 501, 502, 503 are good because the VEIDs are contiguous. A numbering scheme such as 100, 200, 300 is bad because it is noncontiguous. • If you change the VEID, the virtual circuit (VC) reprovisions, and as a result, traffic is impacted.
Step 7	**ve range** *ve-range-number* **Example:** `switch(config-l2vpn-vfi)# ve range`	(Optional) Configures the number of VEIDs for the Autonomous System (AS). • The range for the *ve-range-number* argument is from 1 to 100. The default is 10. • The VE range can be configured based on the number of neighboring PE devices in the network. The VE range value should be approximately the same as the number of neighbors (up to 100). • If no VE range is configured or an existing VE range value is removed, then the default VE range is applied. The default VE range should not be used if the router has many PE neighbors. • If you change the VE range, the virtual circuit (VC) reprovisions and as a result, traffic is impacted.
Step 8	**router bgp** *as-number* **Example:** `switch(config-l2vpn-vfi)# router bgp 100` `switch(config-router)#`	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device. • The *as-number* argument identifies the device to other BGP devices and tags the routing information to be passed along. The range is from 1 to 65535. • The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in the xx.xx format.

	Command	Purpose
Step 9	**bgp graceful restart** **Example:** `switch(config-router)# bgp graceful restart`	Enables the graceful restart and the graceful restart helper capability.
Step 10	**neighbor** *peer-ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 10.1.1.1 remote-as 100`	Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device. • The combination of the *peer-ip-address* and *as-number* arguments must be unique on a device. • The peer IP address is the address of the provider edge (PE) peer. • If the *as-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *as-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor.
Step 11	`address-family l2vpn vpls` **Example:** `switch(config-router)# address-family l2vpn vpls` `switch(config-router-af)#`	Creates an L2VPN address family session and specifies that VPLS endpoint provisioning information is to be distributed to BGP peers.
Step 12	**neighbor** [*peer-ip-address* | *peer-group-name*] **activate** **Example:** `switch(config-router-af)# neighbor 10.10.10.1 activate`	Enables the exchange of information with the specified BGP neighbor
Step 13	**neighbor** [*peer-ip-address* | *peer-group-name*] **send-community extend** **Example:** `switch(config-router-af)# neighbor 10.10.10.1 send-community extend`	Specifies that a community attribute should be sent to the BGP neighbor.
Step 14	**neighbor** [*peer-ip-address* | *peer-group-name*] **suppress-signaling-protocol ldp** **Example:** `switch(config-router-af)# neighbor 10.10.10.1 suppress-signaling-protocol ldp`	Suppresses LDP signaling for a BGP neighbor so that BGP signaling for auto discovery is used.

	Command	Purpose
Step 15	Repeat Steps 11 to 15 to configure additional neighbors in an L2VPN address family.	
Step 16	**copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring BGP Auto Discovery and LDP Signaling

RESTRICTIONS

- You can configure both auto discovered and manually configured pseudowires in a single virtual forwarding instance (VFI). However, the pseudowires cannot go to the same peer PE device.
- You cannot configure a pseudowire by manually configuring a neighbor on one PE device and using auto discovery on the other PE device to configure the same pseudowire in the other direction.
- After enabling VPLS autodiscovery, if you manually configure a neighbor by using the member command and both peers are in autodiscovery mode, each peer will receive discovery data for that VPLS. To prevent peers from receiving data for the VPLS domain, manually configure route target (RT) values. For information, see the "Customizing BGP-Based Auto Discovery Settings (optional)" section.

SUMMARY STEPS

1. **c**onfigure terminal
2. [**no**] **l2vpn vfi context** *vfi-name*
3. (Optional) **description** *description*
4. **vpn** *vpn-id*
5. **autodiscovery bgp signaling ldp**
6. **router bgp** *as-number*
7. **neighbor** *peer-ip-address vc-id*
8. **address-family l2vpn vpls**
9. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **l2vpn vfi context** *vfi-name* **Example:** `switch(config)# l2vpn vfi context foo` `switch(config-l2vpn-vfi)#`	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) context between two or more separate networks. • The *vfi-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 3	**description** *description* **Example:** `switch(config-l2vpn-vfi)# description PWsforVPLS`	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric characters.
Step 4	**vpn** *vpn-id* **Example:** `switch(config-l2vpn-vfi)# mtu 1400`	Configures a Virtual Private Network (VPN) ID on a VFI context. • The valid range is from 1 to 4294967295.
Step 5	**autodiscovery bgp signaling ldp** **Example:** `switch(config-l2vpn-vfi)# autodiscovery bgp signaling ldp`	Enables BGP auto discovery and LDP signaling.
Step 6	**router bgp** *as-number* **Example:** `switch(config-l2vpn-vfi)# router bgp 100` `switch(config-router)#`	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to a device. • The *as-number* argument identifies the device to other BGP devices and tags the routing information to be passed along. the range is from 1 to 65535.

	Command	Purpose
Step 7	**neighbor** *peer-ip-address* **remote-as** *as-number* **Example:** `switch(config-router)# neighbor 10.1.1.1 remote-as 100`	Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device. • The combination of the *peer-ip-address* and *as-number* arguments must be unique on a device. • The peer IP address is the address of the provider edge (PE) peer. • If the *as-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *as-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor.
Step 8	`address-family l2vpn vpls` **Example:** `switch(config-router)# address-family l2vpn vpls`	Creates an L2VPN address family session and specifies that VPLS endpoint provisioning information is to be distributed to BGP peers.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Saves this configuration change.

Customizing BGP-Based Auto Discovery Settings (optional)

Before You Begin

Ensure that you have configured BGP-based auto discovery for VPLS.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **l2vpn vfi context** *vfi-name*
3. (Optional) **vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
4. (Optional) **rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
5. (Optional) **auto-route-target**
   or
   (Optional) **route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **l2vpn vfi context** *vfi-name* **Example:** `switch(config)# l2vpn vfi context foo` `switch(config-l2vpn-vfi)#`	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) context between two or more separate networks. • The *vfi-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 3	**vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `switch(config-l2vpn-vfi)# vpls-id 5:200`	(Optional) Changes the value of the VPLS ID from the generated value to the specified value. • Auto discovery automatically generates a VPLS ID using the Border Gateway Protocol (BGP) autonomous system (AS) number and the configured Virtual Private Network (VPN) ID on the VFI context. • The value for the *nn* argument is the network number.
Step 4	**rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `switch(config-l2vpn-vfi)# rd 2:2`	(Optional) Changes the value of the route distinguisher (RD) from the generated value to the specified value. • Auto discovery automatically generates an RD using the BGP autonomous system number (AS) and the configured Virtual Private Network (VPN) ID on the VFI context. • The value for the *nn* argument is the network number. The network number must be preceded by a colon (:).

	Command or Action	Purpose
Step 5	**auto-route-target** or **route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `switch(config-l2vpn-vfi)# route-target 600:2222`	(Optional) Enables auto discovery to generate a route target (RT) using the lower 6 bits of the RD and the configured Virtual Private Network (VPN) ID on the VFI context. • This is the default. If you previously configured the **route-target** command, use this command to change the explicitly configured RT to a generated RT. or (Optional) Changes the value of the route target (RT) from the generated value to the specified value. • The value for the *nn* argument is the network number. The network number must be preceded by a colon (:).
Step 6	**copy running-config startup-config** **Example:** `switch(config-l2vpn-vfi)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Virtual Private LAN Service with a Bridge Domain

You can configure VPLS either with a bridge domain or with a VLAN. To associate a VFI directly to a VLAN, go to the "Configuring Virtual Private LAN Service with a VLAN" section on page 29-29.

BEFORE YOU BEGIN

- Ensure that you have configured the VFI.
- Ensure that you have configured an EFP for the 802.1Q Access Circuit (AC).

Restrictions

Switchport VLANs and EFPs cannot be associated with the same bridge domain.

SUMMARY STEPS

1. **configure terminal**
2. **feature mpls l2vpn**
3. **feature evc**
4. **system bridge-domain** *id* [**-***id* | **-***id*,...,*id*-*id*]
5. **interface ethernet** *slot/port*
   or
   interface port-channel *port-channel-number*
6. [**no**] **service instance** *service-instance-id* **ethernet**
7. (Optional) **description** description
8. **encapsulation dot1q** *vlan-id*

9. [**no**] **bridge-domain** *domain-id*
10. **member vfi** *vfi-id*
11. **member** *interface slot/port* **service instance** *service-instance-id*
12. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **feature mpls l2vpn** **Example:** `switch(config)# feature mpls l2vpn`	Enables Multiprotocol Label Switching (MPLS) Layer 2 VPN (L2VPN) features. **Note** Using the no feature **mpls l2vpn** command removes all existing L2VPN configurations. Using the feature **mpls l2vpn** command again does not restore the earlier L2VPN configuration.
Step 3	**feature evc** **Example:** `switch(config)# feature evc`	Enables Ethernet virtual circuits on the device.
Step 4	**system bridge-domain** *id* [**-***id* | **-***id***,...,***id***-***id*] **Example:** `switch(config)# system bridge-domain 10-50,100-500`	Identifies the IDs that are available for bridge-domain configurations. • The valid range for the *id* argument is from 2 to 967. • The optional **-***id* keyword and argument combination identifies the last ID in a range of contiguous IDs. The hyphen (-) is required. • The optional list of ID ranges are separated by commas (,). Do not type the ellipses (...).
Step 5	**interface ethernet** *slot/port* or **interface port-channel** *port-channel-number* **Example:** `switch(config)# interface ethernet 2/1` `switch(config-if)#` or `switch(config)# interface port-channel 1` `switch(config-if)#`	Enters interface configuration mode.

	Command	Purpose
Step 6	[**no**] **service instance** *service-instance-id* **ethernet** **Example:** `switch(config-if)# service instance 1 ethernet` `switch(config-if-srv)#`	Enters interface services configuration mode and configures an EFP on the interface. • The *service-instance-id* argument is a unique per-interface identifier for this EFP. The valid range is from 1 to 4000. The range might be restricted due to resource constraints. **Note** You can use the **no** form of this command to delete the EFP and the associated configuration.
Step 7	**description** *description* **Example:** `switch(config-if-srv)# description EFP1forVPLS`	(Optional) Adds a description to this service instance configuration. • The maximum range for the *description* argument is 80 alphanumeric, case-sensitive characters.
Step 8	**encapsulation dot1q** *vlan-id* **Example:** `switch(config-if-srv)# encapsulation dot1q 100`	Allows flow from the specified VLAN ID to pass through the EFP. • The VLAN ID must match the domain ID of the bridge domain to which this EFP is to be associated. The valid range for the *vlan-id* argument is from 2 to 967.
Step 9	[**no**] **bridge-domain** *domain-id* **Example:** `switch(config-if-srv)# bridge-domain 100` `switch(config-bdomain)#`	Enters bridge-domain configuration mode and configures a bridge domain. • The *domain-id* argument is a unique identifier for the bridge domain and underlying VLAN to be created. The valid range is defined by the system bridge-domain configuration. **Note** You can use the **no** form of this command to remove the bridge-domain configuration including port associations. Removing the bridge-domain configuration does not remove the underlying VLAN. If a VLAN is associated with a bridge domain, you cannot remove the VLAN without first removing the bridge domain. To remove the underlying VLAN, use the **no vlan** command after you remove the bridge domain.
Step 10	**member vfi** *vfi-id* **Example:** `switch(config-bdomain)# member vfi foo`	(Optional) Binds a VFI to this bridge domain. • The *vfi-id* argument identifies the VFI to be bound. The maximum range is 100 alphanumeric, case-sensitive characters.

	Command	Purpose
Step 11	**member** *interface slot/port* **service instance** *service-instance-id* **Example:** `switch(config-bdomain)# member ethernet 2/1 service instance 1`	(Optional) Binds a service instance to this bridge domain. • The *interface slot/port* argument identifies the interface under which the service instance is configured. • The *service-instance-id* argument identifies the service instance to be bound. The valid range is from 1 to 4000.
Step 12	**copy running-config startup-config** **Example:** `switch(config-bdomian)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Virtual Private LAN Service with a VLAN

You can configure VPLS either with a bridge domain or with a VLAN. To associate the VFI (or EFP) to a bridge domain, see the "Configuring Virtual Private LAN Service with a Bridge Domain" section on page 29-26.

BEFORE YOU BEGIN

Ensure that you have configured the VFI.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **vlan** vlan-*id*
3. **member vfi** *vfi-id*
4. **exit**
5. **interface ethernet** *slot/port*
6. **switchport mode trunk**
7. **switchport allowed vlan** *vlan-id*
8. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **vlan** *domain-id* **Example:** `switch(config)# vlan 100` `switch(config-vlan)#`	Enters VLAN configuration mode and configures a VLAN. • The *vlan-id* argument is a unique identifier for the VLAN. The valid range is from 1 to 4094. **Note** You can use the **no** form of this command to remove the VLAN configuration including port associations.
Step 3	**member vfi** *vfi-id* **Example:** `switch(config-vlan)# member vfi foo`	Binds a VFI to this VLAN. • The *vfi-id* argument identifies the VFI to be bound. The maximum range is 100 alphanumeric, case-sensitive characters.
Step 4	**exit** **Example:** `switch(config-vlan)# exit` `switch (config)#`	Exits VLAN configuration mode.
Step 5	**interface ethernet** *slot/port* **Example:** `switch(config)# interface ethernet 2/1` `switch(config-if)#`	Enters interface configuration mode and configures an Ethernet interface.
Step 6	**switchport mode trunk** **Example:** `switch(config-if)# switchport mode trunk`	Sets the interface type to be a Layer 2 host port for a trunk.
Step 7	**switchport allowed vlan** *vlan-id* **Example:** `switch(config-if)# switchport allowed vlan 100`	Allows flow from the specified VLAN to pass through the trunk. • The VLAN ID must match the ID of the VLAN to which this VFI is to be associated. The valid range for the *vlan-id* argument is from 1 to 4094.
Step 8	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying the Virtual Private LAN Service Configuration

To verify pseudowire redundancy configuration information, perform one of the following tasks:

Command	Purpose
show ethernet service instance [**detail**]	Displays information about service instances that are configured on the device.
show ethernet service instance interface ethernet *slot/port* [**detail**]	Displays information about service instances that are configured on an interface.
show interface [**brief description**]	Displays the interface status and information.
show interface pseudowire *pw-id*	Displays the status and information about the specified interface.
show interface pseudowire *pw-id* **brief**	Displays brief information about the specified interface.
show interface pseudowire *pw-id* **counters**	Displays the in and out counters for the specified interface. **Note** Multicast and broadcast counters are not supported for pseudowires. All packets and bytes will be counted as unicast.
show interface status	Displays the interface line status.
show interface vfi *name*	Displays the status and information about the specified interface.
show l2vpn atom vc	Displays information about the Any Transport over MPLS (AToM) virtual circuit.
show l2vpn service xconnect all	Displays status information about the specified XConnect service.
show mac address-table	Displays the list of the known MAC addresses and their forwarding information

Monitoring Tunnel Interfaces

You can configure DCNM to collect tunnel interface statistics. Choose **Interfaces > Logical > Tunnel** from the Feature Selector and navigate to the interface that you want to collect statistics on.

You see the Port Traffic Statistics window. You can collect statistics on input and output (packet and byte) counters, broadcast, multicast, and unicast traffic.

See the *Fundamentals Configuration Guide, Cisco DCNM for LAN, Release 5.x* , for more information on collecting statistics for layer 3 interfaces.

Configuration Examples for Virtual Private LAN Service

This section includes the following topics:

Example: VPLS with a Bridge Domain

The following example shows how to configure VPLS with a bridge domain configuration:

```
bridge-domain 100
  member vfi foo
  member Ethernet2/1 service instance 1
!
l2vpn vfi context foo
  vpn id 100
  member Pseudowire12
  member Pseudowire13
!
interface Pseudowire12  #mesh
  encapsulation mpls
  neighbor 10.2.2.2 100
!
interface Pseudowire13  #mesh
  encapsulation mpls
  neighbor 10.3.3.3 100
!
interface Ethernet2/1
   service instance 1 ethernet
      encapsulation dot1q 100
```

Example: VPLS with a VLAN

The following example shows how to configure the same VPLS with a VLAN configuration:

```
vlan 100
vlan configuration 100
  member vfi foo
!
port-profile type pseudowire mpls
  encapsulation mpls
!
l2vpn vfi context foo
  vpn id 100
  member Pseudowire12
  member Pseudowire13
!
interface Pseudowire12 #mesh
  inherit port-profile mpls
  neighbor 10.2.2.2 100
!
interface Pseudowire13  #mesh
  inherit port-profile mpls
  neighbor 10.3.3.3 100
!
interface Ethernet2/1
   switchport mode trunk
```

```
   switchport allowed vlan 100
```

Example: VPLS Auto Discovery and BGP Signaling

The following example show how to configure VPLS auto discovery and BGP signaling:

```
Device bgp 100
    neighbor 10.0.0.2 remote-as 100
      address-family l2vpn vpls
       neighbor 10.0.0.2 activate
       neighbor 10.0.0.2 send-community extended
       neighbor 10.0.0.2 suppress-signaling-protocol ldp
exit-address-family
```

Example: VPLS Auto Discovery and LDP Signaling

The following example show how to configure VPLS auto discovery and LDP signaling:

```
bridge-domain 100
  member vfi foo
  member Ethernet2/1 service instance 1
!
l2vpn vfi context foo
  vpn id 100
  autodiscovery bgp signaling ldp
!
router bgp 100
  neighbor 10.0.0.1 remote-as 100
    address-family l2vpn vpls
!
interface Ethernet2/1
   service instance 1 ethernet
      encapsulation dot1q 100
```

Example: VPLS with MPLS LDP

The following example show how to configure VPLS along with MPLS LDP between PE devices:

PE1

```
feature-set mpls

feature ospf
feature mpls ldp
feature mpls l2vpn

l2vpn
feature evc

vlan 1,100

l2vpn vfi context foo
    vpn id 100
    member 20.0.0.4 encapsulation mpls

vlan configuration 100
    member vfi foo
```

```
interface Ethernet3/1
    switchport
    switchport mode trunk
    switchport trunk allowed vlan 100
    no shutdown

interface Ethernet3/8
    mpls ip
ip address 11.1.1.1/24
    ip router ospf pe1 area 0.0.0.0
    no shutdown

interface loopback0
    ip address 20.0.0.1/32
    ip router ospf pe1 area 0.0.0.0
no terminal log-all
line vty
mpls ldp configuration
    discovery targeted-hello accept
    router-id Lo0 force
    neighbor 20.0.0.4 targeted
router ospf pe1
```

Host P1

```
feature-set mpls
feature ospf
feature mpls ldp

interface Ethernet3/9
    mpls ip
    ip address 11.1.1.2/24
    ip router ospf p1 area 0.0.0.0
    no shutdown

interface Ethernet3/16
    mpls ip
    ip address 12.1.1.1/24
    ip router ospf p1 area 0.0.0.0
    no shutdown

interface loopback0
    ip address 20.0.0.2/32
    ip router ospf p1 area 0.0.0.0

mpls ldp configuration
    router-id Lo0 force
router ospf p1
```

Host P2

```
feature-set mpls
feature ospf
feature mpls ldp

interface Ethernet3/17
    mpls ip
    ip address 12.1.1.2/24
    ip router ospf p2 area 0.0.0.0
    no shutdown
```

```
interface Ethernet3/32
    mpls ip
    ip address 13.1.1.1/24
    ip router ospf p2 area 0.0.0.0
    no shutdown

interface loopback0
    ip address 20.0.0.3/32
    ip router ospf p2 area 0.0.0.0

mpls ldp configuration
    router-id Lo0 force
router ospf p2
```

PE2

```
feature-set mpls
feature ospf
feature mpls ldp
feature mpls l2vpn

l2vpn
feature evc

vlan 1,100

l2vpn vfi context foo
    vpn id 100
    member 20.0.0.1 encapsulation mpls

vlan configuration 100
    member vfi foo

interface Ethernet3/33
    mpls ip
    ip address 13.1.1.2/24
    ip router ospf pe2 area 0.0.0.0
    no shutdown

interface Ethernet3/47
    switchport
    switchport mode trunk
    switchport trunk allowed vlan 100
    no shutdown

interface loopback0
    ip address 20.0.0.4/32
    ip router ospf pe2 area 0.0.0.0
no terminal log-all
line vty
mpls ldp configuration
    discovery targeted-hello accept
    router-id Lo0 force
    neighbor 20.0.0.3 targeted
router ospf pe2
```

Additional References for Virtual Private LAN Service

For additional information related to configuring ACs for VPLS, see the following sections:

Related Documents

Related Topic	Document Title
Interface commands	*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*
VLAN commands	*Cisco Nexus 7000 Series NX-OS Layer 2 Switching Command Reference*
Nondirectly connected MPLS LDP sessions	"Establishing Nondirectly Connected MPLS LDP Sessions" section of the "Configuring the MPLS Label Distribution Protocol" chapter.

MIBs

MIBs	MIBs Link
• BRIDGE-MIB • CISCO-EVC-MIB • CISCO-VLAN-MEMBERSHIP-MIB • CISCO-IETF-PW-ATM-MIB (PW-ATM-MIB) • CISCO-IETF-PW-ENET-MIB (PW-ENET-MIB) • CISCO-IETF-PW-FR-MIB (PW-FR-MIB) • CISCO-IETF-PW-MIB (PW-MIB) • CISCO-IETF-PW-MPLS-MIB (PW-MPLS-MIB) • CISCO-VLAN-MEMBERSHIP-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/dc-os/mibs

Feature History for Virtual Private LAN Service

Table 29-4 lists the release history for this feature.

Table 29-4 Feature History for Virtual Private Lan Service

Feature Name	Releases	Feature Information
Virtual Private Lan Service (VPLS)	8.4(1)	Starting from Cisco NX-OS Release 8.4(1), all VPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on F4-Series I/O modules.
Virtual Private Lan Service (VPLS)	8.2(1)	Starting from Cisco NX-OS Release 8.2(1), all VPLS functionalities, except Ethernet Flow Points (EFP), Service Instances and Bridge Domains, are supported on M3-Series I/O modules.

Table 29-4 Feature History for Virtual Private Lan Service

Feature Name	Releases	Feature Information
Virtual Private Lan Service (VPLS)	6.2(2)	This feature was introduced. The following commands were introduced or modified: **address-family**, **autodiscovery bgp**, **bridge-domain**, **control-word**, **description**, **encapsulation**, **feature mpls l2vpn**, **interface pseudowire**, **l2vpn vfi context**, **member**, **member vfi**, **mtu**, **neighbor**, **router bgp**, **service instance**, **show interface**, **show interface pseudowire**, **show l2vpn atom vc**, **show l2vpn service vfi**, **show l2vpn vfi**, **switchport mode trunk**, **switchport allowed vlan**, **system bridge-domain, vlan**.
IP tunnels in VDC other than default	4.2(1)	This features was introduced.

Configuring Layer 2 VPN Pseudowire Redundancy

This chapter describes how to configure the Layer 2 Virtual Private Network (VPN) Pseudowires Redundancy feature for detecting a failure in the network and rerouting the Layer 2 service to another endpoint that can continue to provide the service.

This chapter includes the following sections:

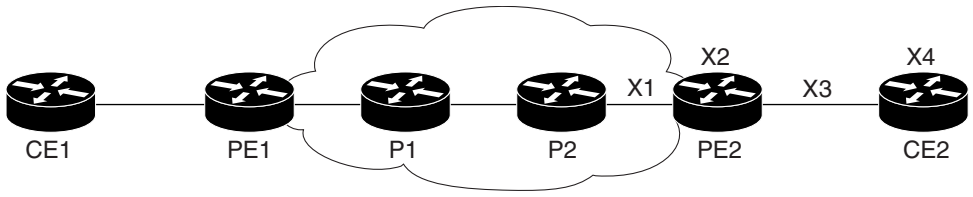
Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About Layer 2 VPN Pseudowire Redundancy

When connectivity between end-to-end provider edge (PE) devices fails, L2VPN pseudowire redundancy can select an alternate path to the directed Label Distribution Protocol (LDP) session and the user data can take over. However, there are some parts of the network where this rerouting mechanism does not protect against interruptions in service. The figure below shows those parts of the network that are vulnerable to an interruption in service.

Figure 30-1 Points of Potential Failure in a Layer 2 VPN Network



X1 = End-to-end routing failure
X2 = PE hardware or software failure
X3 = Attachment circuit failure from a line break
X4 = CE hardware or software failure

The L2VPN Pseudowire Redundancy feature ensures that the customer edge (CE) device, CE2, in the figure above can always maintain network connectivity, even if one or all the failures in the figure occur. When you configure L2VPN pseudowire redundancy, you configure the network with redundant pseudowires (PWs) and redundant network elements.

The figures below show how to set up redundant PWs and Attachment Circuits (ACs) to maintain connectivity.

Figure 30-2 L2VPN Network with Redundant PWs and Attachment Circuits

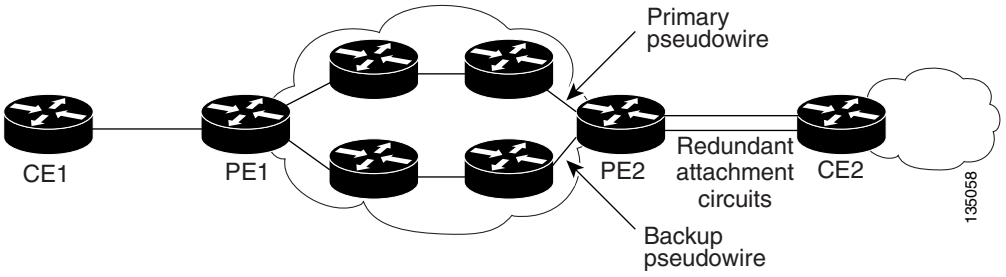


Figure 30-3 L2VPN Network with Redundant PWs, Attachment Circuits, and CE Devices

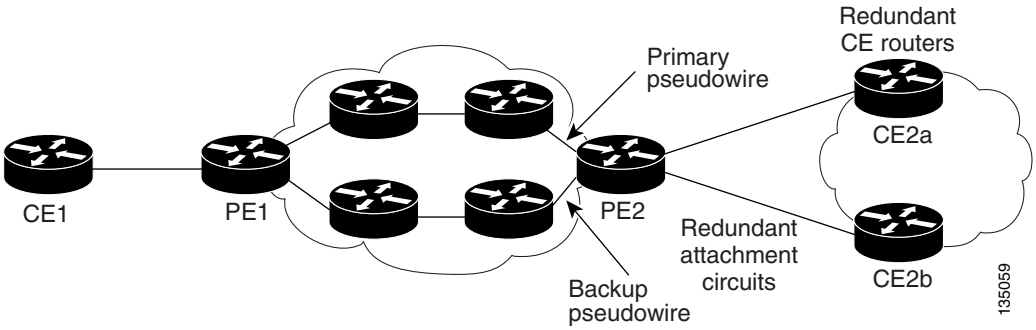
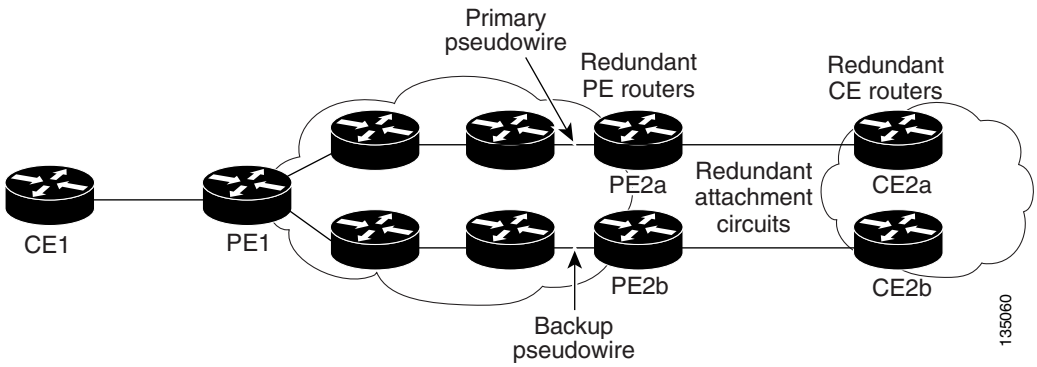


Figure 30-4 L2VPN Network with Redundant PWs, Attachment Circuits, CE Devices, and PE Devices



Licensing Requirements for Layer 2 VPN Pseudowire Redundancy

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco DCNM	IP tunnels require a LAN Enterprise license. For a complete explanation of the DCNM licensing scheme and how to obtain and apply licenses, see the *Cisco DCNM Installation and Licensing Guide, Release 5.x.*
Cisco NX-OS	Layer 2 MVPNs require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Configuring Layer 2 VPN Pseudowire Redundancy

This section includes the following topics:

Configuring a Pseudowire (Optional)

SUMMARY STEPS

1. **configure terminal**
2. **port-profile type pseudowire** *profile-name*
3. **encapsulation mpls**
4. **state enabled**
5. end
6. [**no**] **interface pseudowire** *pw-id*

7. **inherit port-profile** *profile-name*
8. **neighbor** *peer-ip-address vc-id*
9. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**port-profile type pseudowire** *profile-name* **Example:** `switch(config)# port-profile type pseudowire TestSet` `switch(config-if-prof)#`	Enters interface port-profile configuration mode and configures a port profile.
Step 3	**encapsulation mpls** **Example:** `switch(config-if-prof)# encapsulation mpls`	Specifies MPLS encapsulation for this profile.
Step 4	**state enable** **Example:** `switch(config-if-prof)# state enable`	Enables the profile.
Step 5	**end** **Example:** `switch(config-if-prof)# end` `switch(config)`	Returns to privileged EXEC mode.
Step 6	[**no**] **interface pseudowire** *pw-id* **Example:** `switch(config)# interface pseudowire 12` `switch(config-if-pseudowire)#`	Enters interface pseudowire configuration mode and configures a static pseudowire logical interface. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. **Note** You can use the **no** form of this command to delete the pseudowire interface and the associated configuration.
Step 7	**inherit port-profile** *profile-name* **Example:** `switch(config-if-pseudowire)# inherit port-profile TestSet`	Applies a port profile to this interface.

	Command	Purpose
Step 8	**neighbor** *peer-ip-address vc-id* **Example:** `switch(config-if-pseudowire)# neighbor 10.2.2.2 100`	Configures a emulated virtual circuit for this interface. • The combination of the *peer-ip-address* and *vc-id* arguments must be unique on a device. • The peer IP address is the address of the provider edge (PE) peer. • The *vc-id* argument is an identifier for the virtual circuit between devices. The valid range is from 1 to 4294967295.
Step 9	**copy running-config startup-config** **Example:** `switch(config-if-pseudowire)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring a Layer 2 VPN XConnect Context

You can perform this task to add a Layer 2 VPN Attachment Circuit (AC) to associate a backup pseudowire (PW) to the AC.

BEFORE YOU BEGIN

- Ensure that you have configured the AC (Ethernet Flow Point, pseudowire, Ethernet VLAN) for the Layer 2 VPN services.

Restrictions

- There can only be two groups, with a maximum of four members (one as the active and three as backup) in each group, for redundancy.
- If the group name is not specified, only two members can be configured under the Layer 2 VPN XConnect context.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **interface ethernet** *slot*/*port*
3. **no shutdown**
4. **l2vpn xconnect context** *context-name*
5. [**no**] **member** *interface-type slot*/*port* [**service-instance** *service-instance-id*] [**group** *group-name*] [**priority** *number*]
6. [**no**] **member pseudowire** *pw-id* [**group** *name*] [**priority** *number*]
7. [**no**] **member pseudowire** *pw-id* [*peer-addr vc-id* {**encapsulation mpls** | **port-profile** *profile-name*}] [**group** *name*] [**priority** *number*]
8. **redundancy delay** *enable-delay* {*disable-delay* | **never**} **group** *name*
9. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**interface ethernet** *slot*/*port* **Example:** `switch(config)# interface ethernet 2/1` `switch(config-if)#`	Enters interface configuration mode. Make sure that the subinterface on the adjoining CE device is on the same VLAN as this PE device. **Note** You can use the **no** form of this command to delete the interface and the associated configuration.
Step 3	**no shutdown** **Example:** `switch(config-if)# no shutdown`	Brings the port administratively up.
Step 4	[**no**] **l2vpn xconnect context** *context-name* **Example:** `switch(config-if)# l2vpn xconnect context redundancytest` `switch(config-xconnect)#`	Enters Xconnect configuration mode and establishes a Layer 2 VPN (L2VPN) XConnect context for identifying the two members in a Virtual Private Wire Service (VPWS), multisegment pseudowire, or local connect service. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.

	Command	Purpose
Step 5	[**no**] **member** *interface-type slot*/*port* [**service-instance** *service-instance-id*] [**group** *group-name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member ethernet 2/1 service-instance 1 group access-side`	Adds an active Ethernet AC, with or without an Ethernet Flow Point (EFP), to the XConnect context. • The *service-instance-id* argument is a unique per-interface identifier for the EFP. The valid range is from 1 to 4000. The range might be restricted due to resource constraints. • (Optional) The **group** *group-name* keyword and argument combination specifies to which of the redundant groups the member belongs. This configuration is required if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. • You can use the **no** form of this command to delete the specified member configuration.
Step 6	[**no**] **member pseudowire** pw-id [**group** *group-name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member pseudowire 2 group access-side priority 1`	Adds an active pseudowire to the XConnect context. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. • (Optional) The **group** *group-name* keyword and argument combination specifies to which of the redundant groups the member belongs. This configuration is required if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. • You can use the **no** form of this command to delete the specified member configuration.

	Command	Purpose
Step 7	[**no**] **member pseudowire** *pw-id* [*peer-addr vc-id* {**encapsulation mpls** | **port-profile** *profile-name*}] [**group** *name*] [**priority** *number*] **Example:** `switch(config-xconnect)# member pseudowire 3 port-profile TestSet group core priority 1`	(Optional) Creates a backup pseudowire in the XConnect context. This pseudowire configuration is not be displayed in the running configuration and it is not persistent across stateless start ups. • The *pw-id* argument is a unique per-interface identifier for this pseudowire. The range is from 1 to 200000. The range for a static pseudowire is from 1 to 8192. • (Optional) The *peer-address* and *vc-id* arguments configure a emulated virtual circuit for this pseudowire. – The combination of the *peer-ip-address* and *vc-id* arguments must be unique on a device. – The peer IP address is the address of the provider edge (PE) peer. – The *vc-id* argument is an identifier for the virtual circuit between devices. The valid range is from 1 to 4294967295. • (Optional) The **encapsulation mpls** keywords specify MPLS encapsulation for this interface. • (Optional) The **port-profile** and *profile-name* keyword and argument combination specifies that an already-configured pseudowire port profile is to be used for this interface. • (Optional) The **group** *group-name* keyword and argument combination specifies to which of the redundant groups the member belongs. This configuration is required if the member is backed up by one or more other group members in order to identify to which redundant group each member belongs. • (Optional) The **priority** *number* keyword and argument combination specifies the priority of the backup pseudowire in instances where multiple backup pseudowires exist. The range is from 1 to 10, with 1 being the highest priority. The default is 0 and is higher than 1. **Note** You can use the **no** form of this command to delete the specified member configuration.
Step 8	**copy running-config startup-config** **Example:** `switch(config-xconnect)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying the Layer 2 VPN Pseudowire Configuration

To verify pseudowire redundancy configuration information, perform one of the following tasks:

Command	Purpose
show l2vpn atom vc	Displays information about the Any Transport over MPLS (AToM) virtual circuit.
show l2vpn service xconnect all	Displays status information about the specified XConnect service.

Monitoring Tunnel Interfaces

You can configure DCNM to collect tunnel interface statistics. Choose **Interfaces > Logical > Tunnel** from the Feature Selector and navigate to the interface that you want to collect statistics on.

You see the Port Traffic Statistics window. You can collect statistics on input and output (packet and byte) counters, broadcast, multicast, and unicast traffic.

See the *Fundamentals Configuration Guide, Cisco DCNM for LAN, Release 5.x* , for more information on collecting statistics for layer 3 interfaces.

Configuration Examples for Layer 2 Pseudowire Redundancy

The following example shows an Ethernet attachment circuit XConnect and a backup pseudowire:

```
interface Ethernet2/1
  no shutdown
l2vpn xconnect context test
  member pseudowire1 group core
  member 9.9.9.3 200 encapsulation mpls group core priority 2
  member Ethernet2/1
```

The following example shows an Ethernet VLAN attachment circuit XConnect with a service instance and a backup pseudowire:

```
interface Ethernet2/1
  no shutdown
  service instance 100 ethernet
    encapsulation dot1q 100
    no shutdown
l2vpn xconnect context test
  member pseudowire1 group core
  member 9.9.9.3 200 encapsulation mpls group core priority 2
  member Ethernet2/1 service-instance 100
```

The following example shows an Ethernet VLAN attachment circuit XConnect with a subinterface and a backup pseudowire:

```
interface Ethernet2/1.100
  no shutdown
  encapsulation dot1q 100
l2vpn xconnect context test
  member pseudowire1 group core
  member 9.9.9.3 200 encapsulation mpls group core priority 2
  member Ethernet2/1.100
```

Additional References for Layer 2 VPN Pseudowire Redundancy

For additional information related to configuring ACs for VPLS, see the following sections:

- Related Documents, page 30-10

Related Documents

Related Topic	Document Title
Interface commands	*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*
MPLS commands	Cisco Nexus 7000 Series NX-OS MPLS Command Reference

Feature History for Layer 2 VPN Pseudowire Redundancy

Table 30-1 lists the release history for this feature.

Table 30-1 Feature History for Pseudowire Logical Interfaces

Feature Name	Releases	Feature Information
Layer 2 VPN Pseudowire Redundancy	6.2(2)	This feature enables you to set up your network to detect a failure in the network and reroute the Layer 2 service to another endpoint that can continue to provide service. The following commands were introduced or modified: **encapsulation**, **inherit port-profile**, **interface pseudowire**, **l2vpn xconnect context**, **member**, **neighbor**, **port-profile**, **show l2vpn atom vc**, **show l2vpn service xconnect.**
IP tunnels in VDC other than default	4.2(1)	This features was introduced.

Configuring Layer 2 VPN VPLS Dual-Homing with a vPC

This chapter describes how to configure dual-homing with a virtual port channel (vPC) to integrate Virtual Private LAN (VPLS) with the vPC functionality in active-standby mode and allow traffic from a customer edge (CE) device to be load balanced across both provider edge (PE) devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information about Layer 2 VPN VPLS Dual-Homing with a vPC

This section includes the following topics:

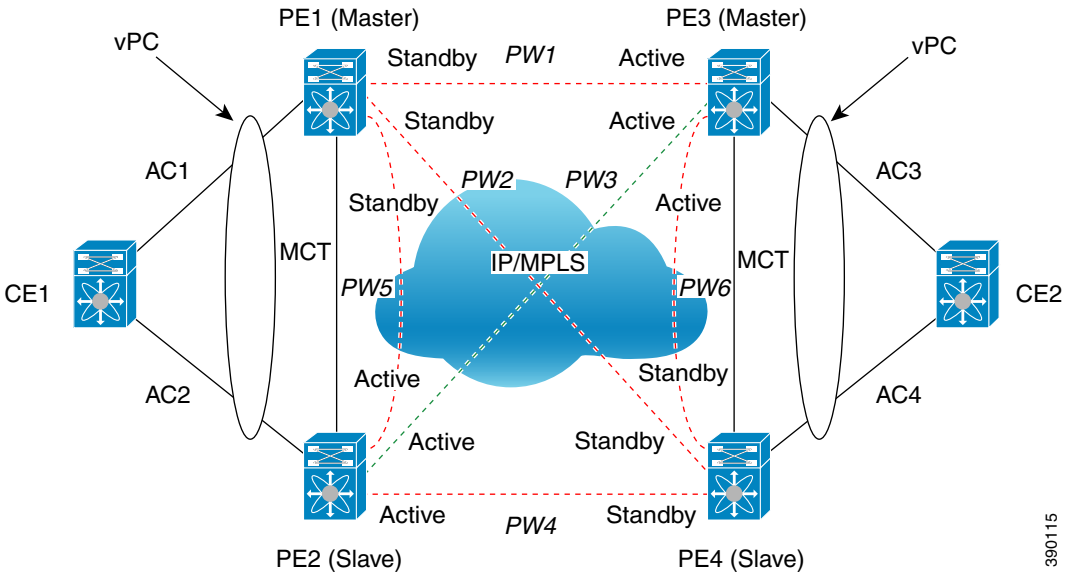
VPLS Integration with vPC

Virtual Private LAN Service (VPLS) provides a multipoint-to-multipoint Layer 2 service over a wide area network (WAN). VPLS is implemented by connecting all the nodes in a particular domain by using a full mesh of pseudowires (PWs).

The virtual port channel (vPC) functionality provides multichassis Ether channel support. Both the attachment circuit (AC) links in a vPC domain are in active mode, which increases the throughput of the network because all the interswitch links can be used to carry traffic.

In the VPLS integration with a vPC, a customer edge (CE) device is dual-homed to two provider edge (PE) devices. The PEs are part of the VPLS domain. One of the PEs in the VPLS domain is in Active state and forwards traffic, while the other PE is in Standby state. See the figure below.

Figure 31-1 VPLS Integration with a vPC



PE1 and PE2 belong to a vPC domain. PE3 and PE4 are part of another vPC domain. In addition to being part of their respective vPC domains, VPLS is configured on the PEs using a mesh of PWs.

In the above figure, the virtual forwarding instance (VFI) configured under a particular VLAN in PE2 is Active for vPC group (PE1, PE2) and VFI configured under a particular VLAN in PE3 is Active for vPC group (PE3, PE4). The Active VFI advertises the local status of Active on all the PWs. The Standby VFI advertises the local status of Standby on all the PWs. A PW is Active when both ends advertise the status of Active. Therefore, PW3 is Active between PE2 and PE3 and is used to carry traffic between CE1 and CE2.

The VPLS domain is configured in decoupled mode. As a result, the status of the AC links is not advertised to the PWs.

Overview of a vPC Peer Link

You can have only two switches as vPC peers; each switch can serve as a vPC peer to only one other vPC peer. The vPC peer switches can also have non-vPC links to other switches.

To create a valid configuration, you must configure an EtherChannel on each switch and then configure the vPC domain. You must assign the EtherChannel on each switch as a peer link. For redundancy, we recommend that you configure at least two dedicated ports into the EtherChannel; if one of the interfaces in the vPC peer link fails, the switch automatically falls back to use another interface in the peer link.

Note The two switches connected by the vPC peer link have certain identical operational and configuration parameters.

When you configure the vPC peer link, the vPC peer switches negotiate that one of the connected switches is the primary switch and the other connected switch is the secondary switch. By default, Cisco NX-OS software uses the lowest MAC address to elect the primary switch. The software takes different actions on each switch only in certain failover conditions. If the primary switch fails, the secondary switch becomes the operational primary switch when the system recovers, and the previously primary switch becomes the secondary switch.

MAC addresses that are learned over vPC links are synchronized between the peers. Configuration information flows across the vPC peer link using the Cisco Fabric Services over Ethernet (CFSoE) protocol. All MAC addresses for VLANs configured on both switches are synchronized between vPC peer switches. The software uses CFSoE for this synchronization.

If the vPC peer link fails, the software checks the status of the remote vPC peer switch by using the peer-keepalive link, which is a link between vPC peer switches, to ensure that both switches are up. If the vPC peer switch is up, the secondary vPC switch disables all vPC ports on its switch. The data then forwards the remaining active links of the EtherChannel. The software learns of a vPC peer switch failure when the keepalive messages are not returned over the peer-keepalive link. Use a separate link (vPC peer-keepalive link) to send configurable keepalive messages between the vPC peer switches. The keepalive messages on the vPC peer-keepalive link determines whether a failure is on the vPC peer link only or on the vPC peer switch. The keepalive messages are used only when all the links in the peer link fail.

Validating the Configuration Between Switches

Cisco NX-OS software validates the configuration between primary (master) and secondary (slave) switches. When a Type-1 mismatch occurs between the primary and secondary switches for a particular VLAN, the VLAN is suspended on both the switches. When a consistency check fails, only the secondary virtual port channel (vPC) switch is brought down. The VLAN remains up on the primary switch and Type-1 configurations can be performed without traffic disruption.

The virtual forwarding instance (VFI) on the vPC switch can be configured as primary or secondary, independent of the vPC state (master or slave) on the switch. Similarly, the VFI on the other vPC switch can be configured as primary or secondary; just not the same as the other vPC peer. If both the vPC peers are configured as primary or secondary or if no primary or secondary vPC peer is configured, a Type-1 error occurs. The table below summarizes the configurations to be avoided.

vPC Peer 1	vPC Peer 2
Primary	Primary
Secondary	Secondary
Primary	—
Secondary	—

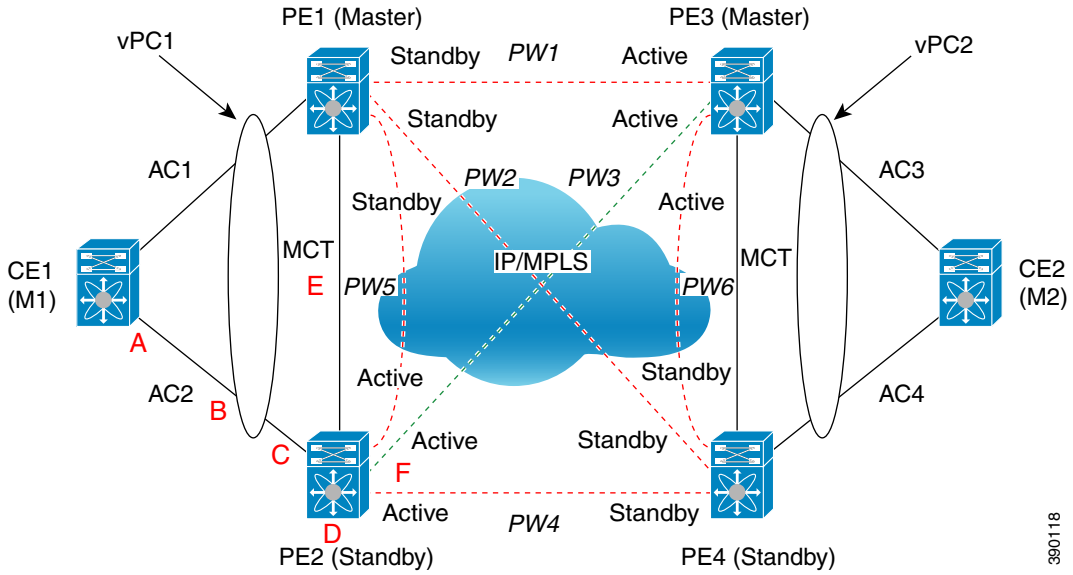
Port, Link, and Node Failures

The VPLS Active-Standby Support with a vPC feature provides network resiliency by protecting against port, link, and node failures. These failures can be categorized into the following scenarios:

- Scenario A: Failure of the uplink port on the dual-homed device (DHD)
- Scenario B: Failure of the uplink (AC) of the DHD
- Scenario C: Failure of the port on a vPC peer
- Scenario D: Failure of the primary node in vPC
- Scenario E: Failure of the vPC peer link (MCT)
- Scenario F: Failure of the vPC node uplink towards the MPLS core

These failure points are shown in the figure below.

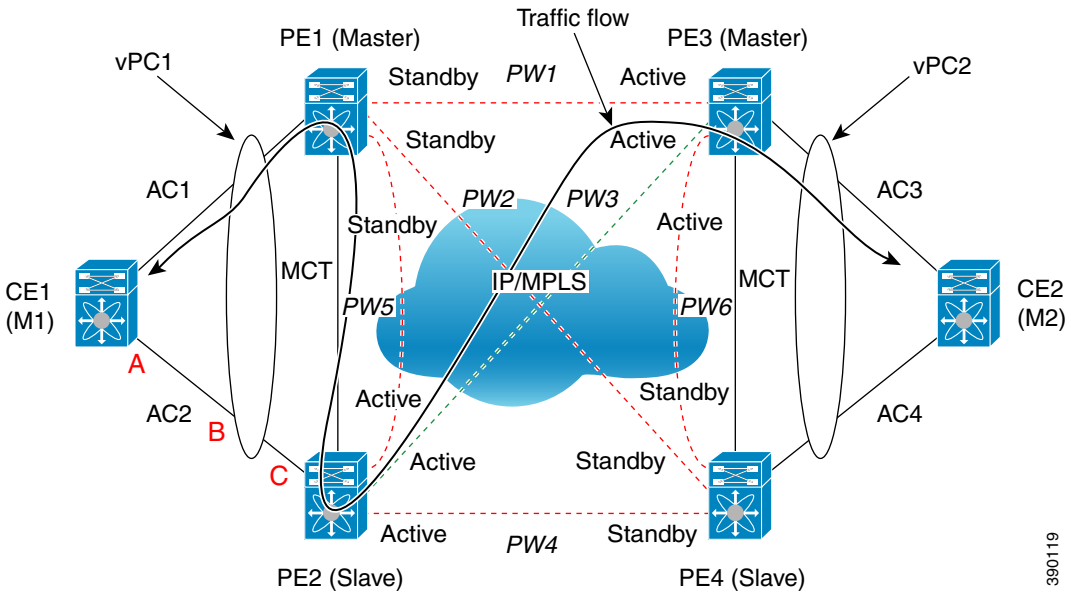
Figure 31-2 Port, Link, and Node Failures



Failed Port or AC Link

The figure below shows the frame flow if a port on the DHD or vPC peer or the attachment circuit (AC) link fails (scenario A, B, or C).

Figure 31-3 Failure Scenario A, B, or C



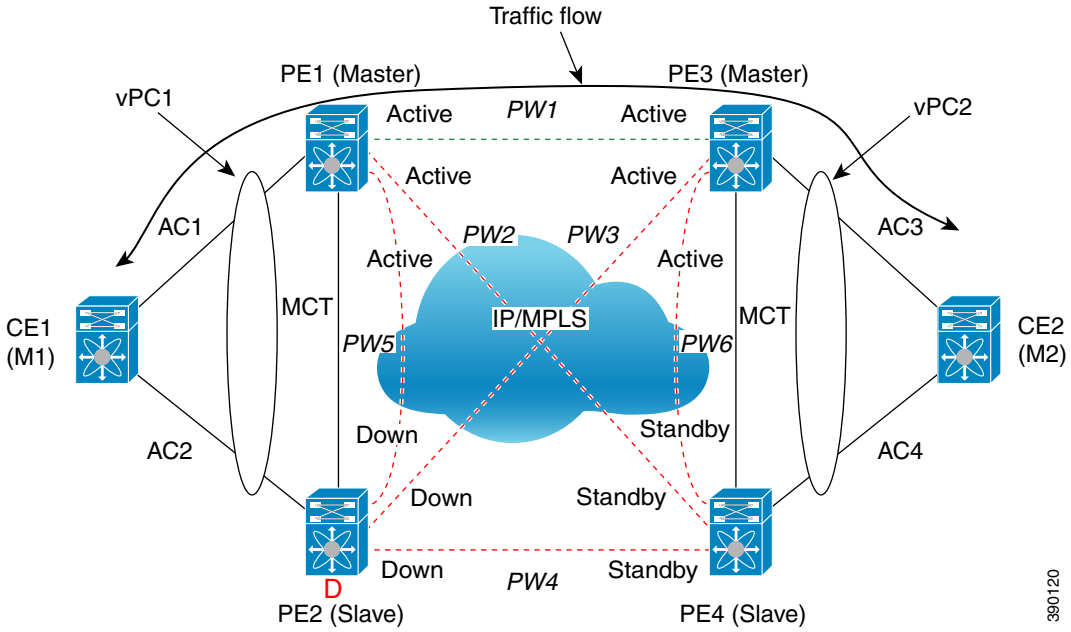
For A, B, or C failures, a vPC diverts the traffic destined from PE2 to AC2 toward multichassis trunk (MCT). In case of A, B, or C failures, PE2 sends a message to PE1 to forward the traffic that is received over MCT to vPC links in addition to sending it over non-vPC links. The traffic is forwarded on AC1 and is received or sent by CE1.

In all of these failure scenarios, because VPLS is configured in decoupled mode, PE2 continues to advertise the local status of Active on all its PWs, and PE1 continues to advertise the local status of Standby.

Failed Primary Node

The figure below shows the switch configured as the primary node fails (scenario D).

Figure 31-4 Failure Scenario D

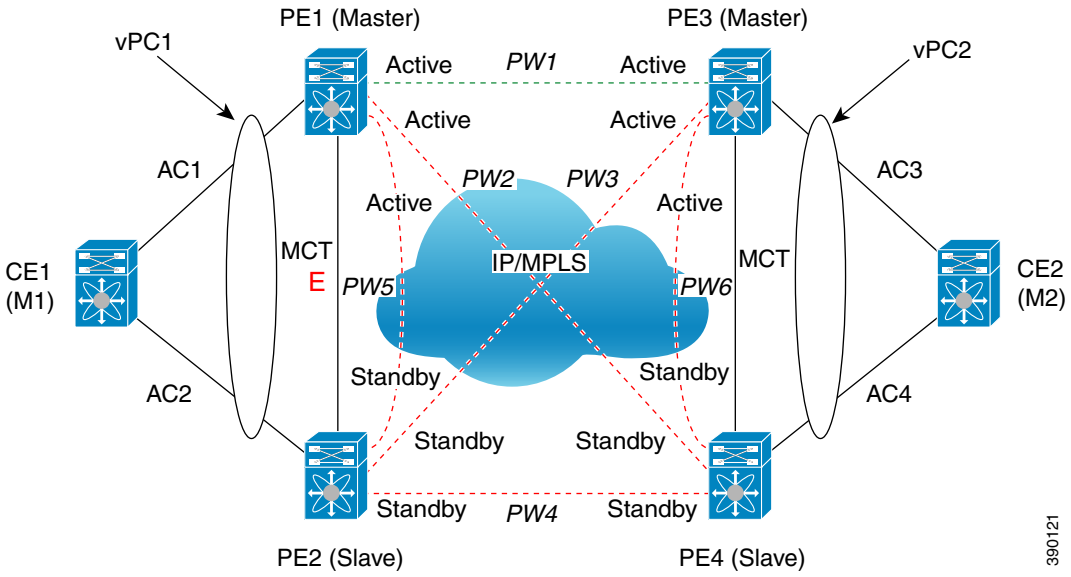


If the primary node in the vPC fails, the peer vPC node detects the failure and starts advertising its local status of Active on all core PWs. PW1 between PE1 and PE3 becomes the Active PW and traffic between vPC1 and vPC2 flows over PW1. If PE3 detects that PW3 to PE2 is down before PE1 has advertised the local status of Active, traffic is dropped by PE3 toward the core (that was using PW3) until PE3 receives the status of Active from PE1.

Failed vPC Peer Link

The figure below shows scenario E or the failure of vPC peer link, also known as multichassis trunk (MCT).

Figure 31-5 Failure Scenario E



If MCT fails and both the nodes in the virtual port channel (vPC) are still up, the vPC master node keeps the AC link up and the vPC slave node brings its AC link down.

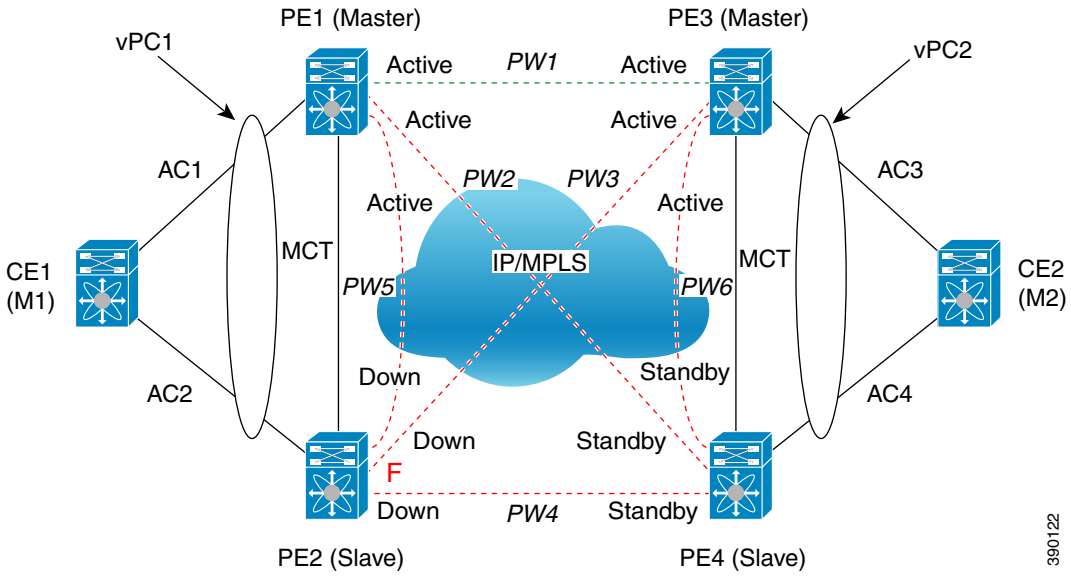
Because VPLS is configured in decoupled mode (the status of the AC link is not advertised to the core pseudowires), the Standby PE2 advertises the local status of Standby to all the core pseudowires (PWs) even though AC2 is down. Therefore, the traffic between vPC1 and vPC2 flows over PW1. The traffic between CE1 and PE1 flows only through the AC1 link.

You can configure vPC to detect a double fault if both MCT and the primary vPC node fail. You can configure vPC by using the out-of-band keepalive mechanism. In this scenario, the secondary vPC node keeps the AC link (AC) up and PE2 continues to advertise the status of Active to all core PWs. PW3 remains active.

Failed Core Pseudowires

The figure below shows the failure of core pseudowires (PWs) on the Active vPC node (scenario F).

Figure 31-6 Failure Scenario F



When all PWs in the core on the Active vPC node go down, the peer Standby vPC node changes its state to Active and advertises the local status of Active on all core PWs. PW 1 becomes Active.

When all core interfaces on a node go down, the PWs in the VPLS domain also go down.

Licensing for Layer 2 VPN VPLS Dual-Homing with a vPC

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Layer 2 MVPN requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations for Layer 2 VPN VPLS Dual-Homing with a vPC

The VPLS Dual-Homing with a vPC feature is supported only on switch port links. Virtual port channel (vPC) is not supported on an Ethernet virtual circuit (EVC) port.

Configuring Layer 2 VPN VPLS Dual-Homing with a vPC

You can configure a vPC peer as the primary node in a dual-homed topology. Repeat this task to configure the other vPC peer as the secondary node.

Before You Begin

- Ensure that the Layer 2 VPN feature is enabled on the switch.
- Ensure that the dual-homed vPC domains are configured.
- Ensure that a VPLS domain with a mesh of core PWs that connect the PEs of the vPC domain is configured.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **l2vpn vfi context** *vfi-name*
3. (Optional) **description** *description*
4. **vpn id** *vpn-id*
5. **redundancy** {**primary** | **secondary**}
6. **member** *ip-address* **encapsulation mpls**
7. **exit**
8. [**no**] **bridge domain** *domain-id*
9. **member vfi** *vfi-name*
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	[**no**] **l2vpn vfi context** *vfi-name* **Example:** `switch(config)# l2vpn vfi context vpls80` `switch(config-l2vpn-vfi)#`	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) between two or more separate networks. • The *vfi-name* argument is a unique per-interface identifier for this VFI. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the VFI and the associated configuration.
Step 3	**description** *description* **Example:** `switch(config-l2vpn-vfi)# description VFIforDualHome`	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric characters.

	Command	Purpose
Step 4	**vpn** *vpn-id* **Example:** `switch(config-l2vpn-vfi)# vpn`	Configures a Virtual Private Network (VPN) ID on a VFI context. • The valid range is from 1 to 4294967295.
Step 5	**redundancy** {**primary** | **secondary**} **Example:** `switch(config-l2vpn-vfi)# redundancy primary`	Configures this L2VPN VFI context as the primary or secondary node.
Step 6	**member** *ip-address* **encapsulation mpls** **Example:** `switch(config-l2vpn-vfi)# member 10.0.0.3 encapsulation mpls`	Specifies the devices that form a point-to-point L2VPN VFI connection.
Step 7	**exit** **Example:** `switch(config-l2vfi-vfi)# vpn 80` `switch (config)#`	Exits Layer 2 VFI configuration mode.
Step 8	[**no**] **bridge-domain** *domain-id* **Example:** `switch(config)# bridge-domain 100` `switch(config-bdomain)#`	Enters bridge-domain configuration mode and configures a bridge domain. • The *domain-id* argument is a unique identifier for the bridge domain and underlying VLAN to be created. The valid range is defined by the system bridge-domain configuration. **Note** You can use the **no** form of this command to remove the bridge-domain configuration including port associations. Removing the bridge-domain configuration does not remove the underlying VLAN. If a VLAN is associated with a bridge domain, you cannot remove the VLAN without first removing the bridge domain. To remove the underlying VLAN, use the **no vlan** command after you remove the bridge domain.
Step 9	**member vfi** *vfi-name* **Example:** `switch(config-bdomain)# member vfi vpls80`	(Optional) Binds a VFI to this bridge domain. • The *vfi-name* argument identifies the VFI to be bound. The maximum range is 100 alphanumeric, case-sensitive characters.
Step 10	**copy running-config startup-config** **Example:** `switch(config-if-srv)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuration Examples for Layer 2 VPN VPLS Dual-Homing with a vPC

PE1

```
l2vpn vfi context vpls-80
  vpn id 80
  redundancy primary
  member 10.0.0.4 encapsulation mpls
!
bridge-domain 80
  member vfi vpls-80
```

PE2

```
l2vpn vfi context vpls-80
  vpn id 80
  redundancy secondary
  member 10.0.0.4 encapsulation mpls
!
bridge-domain 80
  member vfi vpls-80
```

Additional References for Layer 2 VPN VPLS Dual-Homing with a vPC

Related Documents

Related Topic	Document Title
Interface commands	*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*
VLAN commands	*Cisco Nexus 7000 Series NX-OS Layer 2 Switching Command Reference*
Virtual port channels	"Configuring vPCs" chapter of the *Cisco Nexus 7000 Series NX-OS Interfaces Configuration Guide*

Feature History for Layer 2 VPN VPLS Dual-Homing with a vPC

Table 31-1 lists the history for this feature.

Table 31-1 Feature History for Layer 2 VPN VPLS Dual-Homing wit vPC

Feature Name	Releases	Feature Information
VPLS Dual-Homing with a vPC	6.2(2)	The VPLS Dual-Homing with a vPC feature integrates Virtual Private LAN (VPLS) with the virtual port channel (vPC) functionality in active-standby mode. This feature allows traffic from a customer edge (CE) device to be load-balanced across both provider edge (PE) devices. The active PE can then forward the traffic to the core. Similarly, traffic from the core can be received by the active PE and sent to the attached CE.
IP tunnels in VDC other than default	4.2(1)	This features was introduced.

Configuring MVPNs

This chapter describes how to configure multicast virtual private networks (MVPNs) on Cisco NX-OS devices.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MVPNs

You can use an MVPN feature to support multicast over a Layer 3 VPN. IP multicast is used to stream video, voice, and data to an VPN network core.

Historically, point-to-point tunnels were the only way to connect through an enterprise or service provider network. Although such tunneled networks had scalability issues, they were the only means of passing IP multicast traffic through a virtual private network (VPN).

Because Layer 3 VPNs support only unicast traffic connectivity, deploying with a Layer 3 VPN allows operators to offer both unicast and multicast connectivity to Layer 3 VPN customers.

This section includes the following topics:

MVPN Overview

An MVPN allows an operator to configure and support multicast traffic in an MVPN environment. MVPNs support routing and forwarding of multicast packets for each individual virtual routing and forwarding (VRF) instance, and it also provides a mechanism to transport VPN multicast packets across the enterprise or service provider backbone. IP multicast is used to stream video, voice, and data to a VPN network core.

A VPN allows network connectivity across a shared infrastructure, such as an Internet Service Provider (ISP). Its function is to provide the same policies and performance as a private network at a reduced cost of ownership.

MVPNs allow an enterprise to transparently interconnect its private network across the network backbone. Using MVPNs to interconnect an enterprise network does not change the way that an enterprise network is administered and it does not change general enterprise connectivity.

MVPN Routing and Forwarding and Multicast Domains

MVPNs introduce multicast routing information to the VPN routing and forwarding table. When a provider edge (PE) router receives multicast data or control packets from a customer edge (CE) router, the router forwards the data or control packets according to the information in the MVPN routing and forwarding (MVRF). MVPNs do not use label switching.

A set of MVRFs that can send multicast traffic to each other constitutes a multicast domain. For example, the multicast domain for a customer that wanted to send certain types of multicast traffic to all global employees would consist of all CE routers that are associated with that enterprise.

Multicast Distribution Trees

MVPNs establish a static default multicast distribution tree (MDT) for each multicast domain. The default MDT defines the path used by PE routers to send multicast data and control messages to every other PE router in the multicast domain.

MVPNs also support the dynamic creation of MDTs for high-bandwidth transmission. Data MDTs are intended for high-bandwidth sources such as full-motion video inside the VPN to ensure optimal traffic forwarding in the VPN core. When the multicast transmission exceeds the defined threshold, the sending PE router creates the data MDT and sends a User Datagram Protocol (UDP) message, which contains

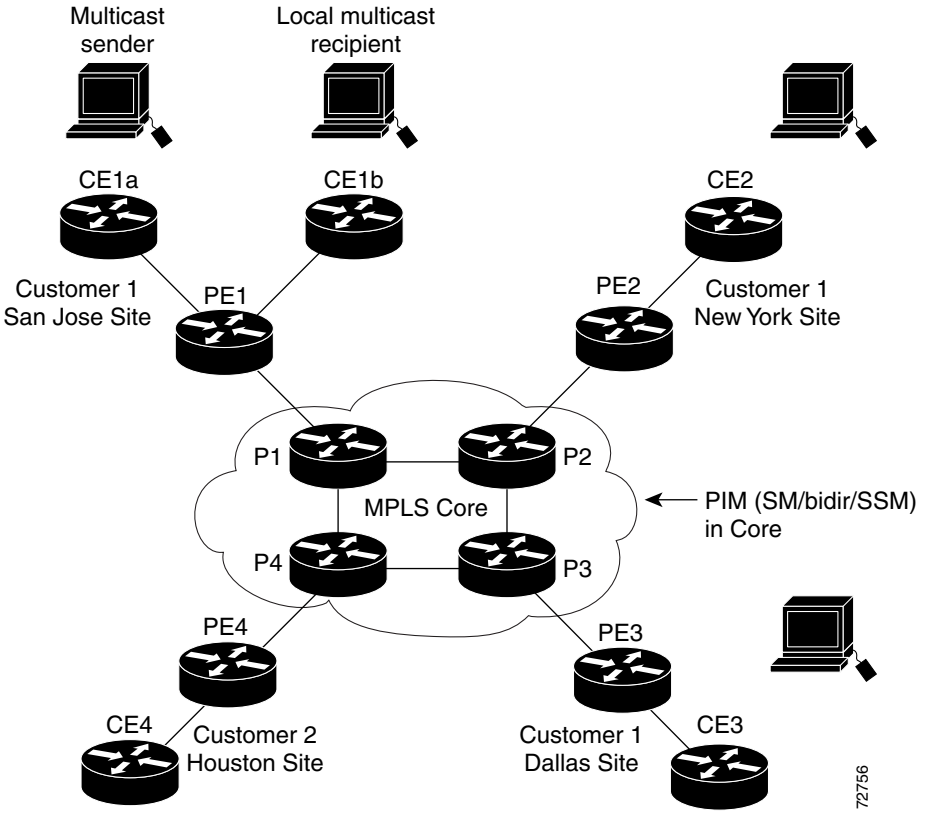
information about the data MDT, to all routers on the default MDT. Once every second, the PE router examines the statistics to determine whether a multicast stream has exceeded the data MDT threshold. After a PE router sends the UDP message, it waits 3 more seconds before switching over.

Data MDTs are created for bidirectional routes if you use the **mdt data bidir-enable** command in that VRF. (Data MDTs are not created for bidirectional customer routes by default.)

In the following example, a service provider has a multicast customer with offices in San Jose, New York, and Dallas. A one-way multicast presentation is occurring in San Jose. The service provider network supports all three sites that are associated with this customer, in addition to the Houston site of a different enterprise customer.

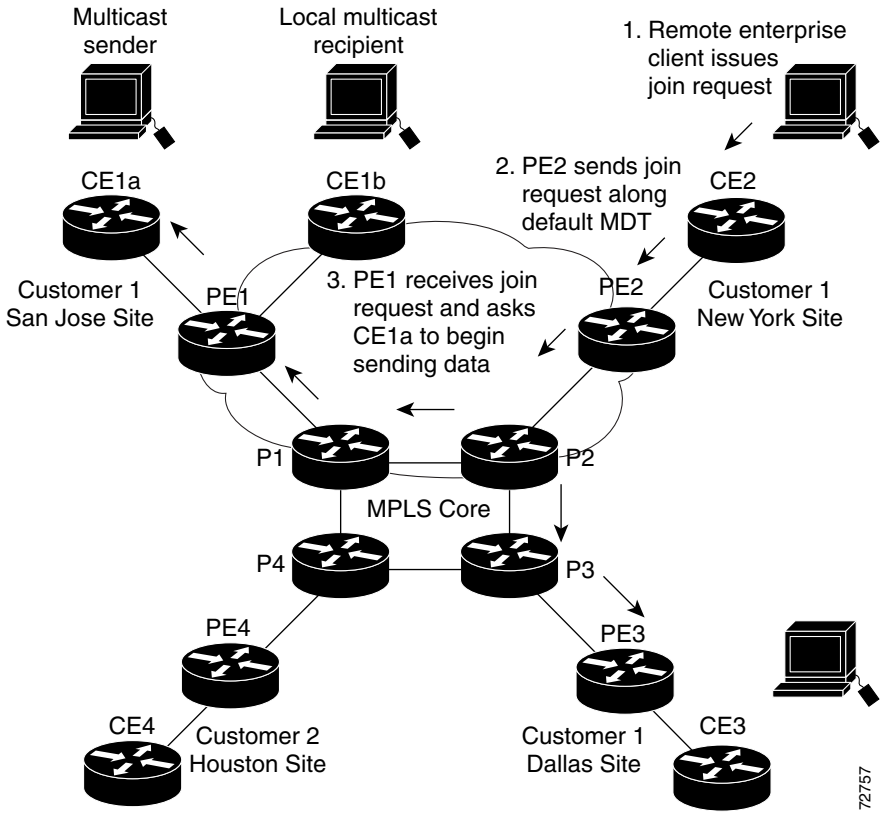
The default MDT for the enterprise customer consists of provider routers P1, P2, and P3 and their associated PE routers. PE4 is not part of the default MDT, because it is associated with a different customer. Figure 32-1 shows that no data flows along the default MDT, because no one outside of San Jose has joined the multicast.

Figure 32-1 Default Multicast Distribution Tree Overview



An employee in New York joins the multicast session. The PE router that is associated with the New York site sends a join request that flows across the default MDT for the multicast domain of the customer. PE1, the PE router that is associated with the multicast session source, receives the request. Figure 32-2 depicts that the PE router forwards the request to the CE router that is associated with the multicast source (CE1a).

Figure 32-2 Initializing the Data MDT



The CE router (CE1a) begins to send the multicast data to the associated PE router (PE1), which sends the multicast data along the default MDT. Immediately after sending the multicast data, PE1 recognizes that the multicast data exceeds the bandwidth threshold for which a data MDT should be created. Therefore, PE1 creates a data MDT, sends a message to all routers using the default MDT that contains information about the data MDT, and, three seconds later, begins sending the multicast data for that particular stream using the data MDT. Only PE2 has interested receivers for this source, so only PE2

joins the data MDT and receives traffic on it. (If the data MDT had not been configured and only the default MDT had been configured, all the customer sites would have received the traffic even though they were not interested in it.)

PE routers maintain a PIM relationship with other PE routers over the default MDT and a PIM relationship with its directly attached P routers.

Multicast Tunnel Interface

An MVPN routing and forwarding (MVRF), which is created per multicast domain, requires the router to create a tunnel interface from which all MVRF traffic is sourced. A multicast tunnel interface is an interface that the MVRF uses to access the multicast domain. The interface is a conduit that connects an MVRF and the global MVRF. One tunnel interface is created per MVRF.

Benefits of MVPNs

The benefits of MVPNs are as follows:

- Provides a scalable method to dynamically send information to multiple locations
- Provides high-speed information delivery
- Provides connectivity through a shared infrastructure

Information About the BGP Advertisement Method for MVPN Support

This section includes the following topics:

Overview

When you configure the default MDT in a PIM Source Specific Multicast (PIM-SSM) environment rather than a PIM-SM environment, the receiver PE needs information about the source PE and the default MDT. This information is used to send (S, G) joins toward the source PE to build a distribution tree from the source PE without the need for a rendezvous point (RP). The source provider edge (PE) address and default MDT address are sent using the Border Gateway Protocol (BGP).

BGP MDT SAFI

BGP MDT SAFI is the BGP advertisement method that is used for MVPNs. In the current release, only IPv4 is supported. MDT SAFI has the following settings:

- AFI = 1
- SAFI = 66

In Cisco NX-OS, the source PE address and the MDT address are passed to PIM using BGP MDT SAFI updates. The Route Descriptor (RD) type has changed to RD type 0 and BGP determines the best path for the MDT updates before passing the information to PIM.

You must configure the MDT SAFI address family for BGP neighbors by using the **address-family ipv4 mdt** command. You must still enable neighbors that do not support the MDT SAFI for the MDT SAFI in the local BGP configuration. Prior to the MDT SAFI, additional BGP configuration from the VPNv4 unicast configuration was not needed to support MVPNs.

Licensing Requirements for MVPNs

Product	License Requirement
Cisco NX-OS	*MVPNs require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the Cisco NX-OS Licensing Guide.*

Prerequisites for MVPNs

Configuring MVPNs has the following prerequisites:

- Ensure that you have configured MPLS and Label Distribution Protocol (LDP) in your network. All routers in the core, including the PE routers, must be able to support MPLS forwarding. VPNv4 routes are not installed by BGP if labeled paths do not exist for PE source addresses.
- Ensure that you have installed the correct license for MPLS and any other features you will be using with MPLS.

Guidelines and Limitations for MVPNs

MVPNs have the following configuration guidelines and limitations:

- Bidirectional Forwarding Detection (BFD) is not supported on the Multicast Tunnel Interface (MTI).
- By default, the BGP update source is used as the source of the MVPN tunnel. However, you can use the **mdt source** to override the BGP update source and provide a different source to the multicast tunnel.
- Cisco NX-OS Release 5.2(4) and later 5.x releases as well as Cisco NX-OS Release 6.1(1) and later 6.x releases support multicast GRE tunnel interfaces for PE-CE routing with MVPN.
- In Cisco Nexus 7000 Series, you can have up to 30 seconds of duplication upon P-PE link flap with default CoPP values. CoPP for RPF failure packets needs to be disabled to reduce the duplication window. By applying the following CoPP profile, duplication window can be reduced.

```
switch# copp copy profile strict prefix xyz
switch(config)# class-map type control-plane match-any xyz-copp-class-normal
switch(config-cmap)# match exception ip multicast rpf-failure
switch(config-cmap)# match exception ipv6 multicast rpf-failure
switch(config-cmap)# control-plane
switch(config-cp)# service-policy input xyz-copp-policy-strict
switch(config-cp)# end
```

MDT SAFI has the following configuration and limitations guidelines:

- You must configure the MDT SAFI on all routers that participate in the MVPN operations.
- Extended communities are needed for VPNv4 interior BGP (iBGP) sessions to carry the connector attribute.

Default Settings for MVPNs

Table 32-1 lists the default settings for MVPN parameters.

Table 32-1 Default MVPN Parameters

Parameters	Default
mdt default *address*	No default
mdt enforce-bgp-mdt-safi	Enabled
mdt data *threshold*	0 Kilobits/second
mdt source	No default
mdt mtu *mtu*[1]	1376 bytes
mdt ip pim hello-interval *interval*	30000 ms
mdt ip pim jp-interval *interval*	60000 ms
mdt data bidir-enable[2]	Disabled
mdt default asm-use-shared-tree [only][3]	Disabled

1. The default MDT MTU value for Cisco Catalyst 6000 Series switches is 1500 bytes, which is different from the default value of 1376 bytes for Cisco Nexus 7000 Series switches. To avoid an interoperability issue (especially when migrating from the Cisco Catalyst 6000 Series switches), make sure to use the appropriate MDT MTU value.
2. Enables data MDTs to be created for bidir customer routes.
3. The receiving PE's do not trigger an (S,G) join toward the source for the MDT routes when default MDT is in PIM ASM mode.

Configuring MVPNs

This section includes the following topics:

Enabling Features

You enable required features by using the detailed steps in this section. This procedure is required for enabling features.

Note Some protocols, such as rip/ospf, must be running both on customer VRFs as well as the core.

SUMMARY STEPS

1. **configure terminal**
2. feature bgp
3. feature pim
4. feature mvpn
5. feature mpls l3vpn
6. feature mpls ldp

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**feature pim** **Example:** `switch(config)# feature pim`	Enables the PIM feature.
Step 4	**feature mvpn** **Example:** `switch(config)# feature mvpn`	Enables the MVPN feature.
Step 5	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature, which is needed to determine unicast routes across sites.
Step 6	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS Label Distribution Protocol (LDP).

Enabling PIM on Interfaces

You can configure Protocol Independent Multicast (PIM) on all interfaces that are used for IP multicast. We recommend that you configure PIM sparse mode on all physical interfaces of provider edge (PE) routers that connect to the backbone. We also recommend that you configure PIM sparse mode on all loopback interfaces if they are used for BGP peering or if their IP address is used as an RP address for PIM.

Note This procedure is required for enabling PIM on interfaces. For more information on PIM, see the *Cisco Nexus 7000 Series NX-OS Multicast Routing Configuration Guide*.

SUMMARY STEPS

1. configure terminal
2. **ip pim sparse-mode**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**ip pim sparse-mode** **Example:** `switch (config-if)# ip pim sparse-mode`	Enables PIM sparse mode on the interface.

Configuring a Default MDT for a VRF

You can configure a default MDT for a VRF.

The default MDT must be the same that is configured on all routers that belong to the same VPN. The source IP address is the address that you use to source the BGP sessions.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **mdt default** *address*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**vrf context** *vrf-name* **Example:** `switch(config)# vrf context vrf1`	Sets the VRF context by assigning a VRF name.
Step 3	**mdt default** *address* **Example:** `switch(config-vrf)# mdt default 232.0.0.1`	Configures the multicast address range for data MDTs for a VRF as follows: • A tunnel interface is created as a result of this command. • By default, the destination address of the tunnel header is the *address* argument.

Enforcing MDT SAFI for a VRF

You can enforce the use of MDT subsequent address family identifiers (SAFI) for a VRF, or you can configure MDT to interoperate with peers that do not support MDT SAFI.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **[no] mdt enforce-bgp-mdt-safi**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal	Enters global configuration mode.
Step 2	**vrf context** *vrf-name* **Example:** switch(config)# vrf context vrf1 switch(config-vrf)#	Sets the VRF context by assigning a VRF name.
Step 3	**[no] mdt enforce-bgp-mdt-safi** **Example:** switch(config-vrf)# mdt enforce-bgp-mdt-safi	Enforces the use of MDT SAFI for the specified VRF. The **no** form of this command enables MDT to interoperate with peers that do not support MDT SAFI. When the **no** form is used, initially only the (*,G) entry for the default MDT group is populated if it falls within the Any Source Multicast (ASM) range. Then later, based on traffic, the (S,G) entries are learned like regular ASM routes.

Configuring the MDT Address Family in BGP for MVPNs

You can configure an MDT address family session on PE routers to establish MDT peering sessions for MVPNs.

Use the **address-family ipv4 mdt** command under neighbor mode to configure an MDT address-family session. MDT address-family sessions are used to pass the source PE address and MDT address to PIM using BGP MDT Subaddress Family Identifier (SAFI) updates.

Prerequisites

Before MVPN peering can be established through an MDT address family, you must configure MPLS in the BGP network and multiprotocol BGP on PE routers that provide VPN services to CE routers.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp** ***as-number***
3. **vrf context** *vrf-name*
4. **rd** *route-distinguisher*
5. **address-family ipv4** *unicast*
6. **route-target import** *route-target-ext-community*
7. **route-target export** *route-target-ext-community*
8. **router bgp** *as-number*
9. **address-family ipv4 mdt**
10. **address-family {vpnv4} [unicast]**
11. **address-family {ipv4} [unicast]**
12. **neighbor** *neighbor-address*

13. **update source** *interface*
14. **address-family ipv4 mdt**
15. **address-family vpnv4** [**unicast**]
16. **send-community extended**
17. (Optional) **show bgp** {**ipv4**} **unicast neighbors vrf** *vrf-name*
18. (Optional) copy running-config startup-config

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal` `switch(config)#`	Enters global configuration mode.
Step 2	**feature bgp** *as-number* **Example:** `switch(config)# feature bgp 65535`	Enters switch configuration mode and creates a BGP routing process.
Step 3	**vrf context** *vrf-name* **Example:** `switch(config)# vrf context vpn1` `switch(config-vrf)#`	Defines a VPN routing instance identified by *vrf-name* and enters VRF configuration mode. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 4	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd 1.2:1`	Assigns a route distinguisher to the VRF *vrf-name*. The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter an RD in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 5	**address-family ipv4 unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast` `switch(config-vrf-af)#`	Specifies the IPv4 address family type and enters address family configuration mode.
Step 6	**route-target import** *route-target-ext-community* **Example:** `switch(config-vrf-af)# route-target import 1.0:1`	Specifies a route-target extended community for a VRF. The **import** keyword imports routing information from the target VPN extended community. The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF list of import route-target extended communities. You can enter the *route-target-ext-community* argument in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 – 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1

	Command	Purpose
Step 7	**route-target export** *route-target-ext-community* **Example:** `switch(config-vrf-af)# route-target export 1.0:1`	Specifies a route-target extended community for a VRF. The **export** keyword exports routing information to the target VPN extended community. The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF list of export route-target extended communities. You can enter the *route-target-ext-community* argument in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 8	**router bgp** *as-number* **Example:** `switch(config)# router bgp 1.1` `switch(config-router)#`	Configures a BGP routing process and enters router configuration mode. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 9	**address-family ipv4 mdt** **Example:** `switch(config-router)# address-family ipv4 mdt`	Enters IPv4 MDT address family configuration mode.
Step 10	**address-family {vpnv4} [unicast]** **Example:** `switch(config-router-af)# address-family vpnv4` `switch(config-router-af)#`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard VPNv4 or VPNv6 address prefixes. The optional **unicast** keyword specifies VPNv4 or VPNv6 unicast address prefixes.
Step 11	**address-family** {**ipv4**} **unicast** **Example:** `switch(config-router-af)# address-family ipv4 unicast` `switch(config-router-af)#`	Enters address family configuration mode for configuring routing sessions that use standard IPv4 or IPv6 address prefixes.
Step 12	**neighbor** *neighbor-address* **Example:** `switch(config-switch-af)# neighbor 192.168.1.1`	Enters neighbor configuration mode.
Step 13	**update source** *interface* Example: `switch (config-router-neighbor)# update-source loopback 1`	Sets the update source as loopback1.
Step 14	**address-family ipv4 mdt** **Example:** `switch(config-router-neighbor)# address-family ipv4 mdt`	Enters address family configuration mode to create an IP MDT address family session.

	Command	Purpose
Step 15	**address-family vpnv4 [unicast]** **Example:** switch(config-router-neighbor-af)# address-family vpnv4 switch(config-router-neighbor-af)#	Enters VPNv4 address family configuration mode.
Step 16	**send-community extended** **Example:** switch(config-router-neighbor-af)# send-community extended	Specifies that extended communities attribute should be sent to a BGP neighbor.
Step 17	**show bgp** {**ipv4**} **unicast neighbors vrf** *vrf-name* **Example:** switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors vrf vpn1	(Optional) Displays information about BGP neighbors. The *vrf-name* argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 18	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring a Data MDT

You can configure a data MDT.

Multicast groups that are used to create the data MDT are dynamically chosen from a pool of configured IP addresses. If the number of streams is greater than the maximum number of data MDTs per VRF per PE, multiple streams share the same data MDT. See Appendix A, "Configuration Limits for Cisco NX-OS MPLS" for information on the maximum supported number of data MDTs per VRF per PE.

Prerequisites

Before configuring a data MDT, you must configure the default MDT on the VRF.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **mdt data data** *prefix* [**threshold** *threshold-value*] [**routemap** *policy-name*]
4. **exit**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal	Enters global configuration mode.
Step 2	**vrf context** *vrf-name* **Example:** switch(config)# ip vrf vrf1	Enters VRF configuration mode and defines the VPN routing instance by assigning a VRF name.
Step 3	**mdt data data prefix** *[**threshold** threshold-value] [**routemap** policy-name]* **Example:** switch(config-vrf)# mdt data 232.7.7.0/24 threshold 10 route-map rmap2mdt data 239.192.20.32 0.0.0.15 threshold 1	Specifies a range of threshold values as follows: • *Prefix* specifies the range of addresses to be used in the data MDT pool. • *Threshold-value* specifies the threshold in kilobits per second when the stream is switched to the data MDT. • *Policy-name* defines a policy file that defines which customer data streams should be considered for switching onto the data MDT.
Step 4	**exit** **Example:** switch(config-vrf)# exit	Returns to global configuration mode.

Verifying the MVPN Configuration

To display the MVPN configuration, perform one of the following tasks:

Command	Purpose
show interface	Displays details of an interface.
show ip mroute vrf	Displays multicast routes.
show ip pim event-history mvpn	Displays the details of the MVPN event history logs.
show ip pim mdt	Displays the details of MTI tunnels created by MVPN.
show ip pim mdt receive	Displays the mapping of the customer source, the customer group to data MDT source, and the data MDT group on the receiving side.
show ip pim mdt send	Displays the mapping of the customer source, the customer group to data MDT source, and the data MDT group on the sending side.
show ip pim neighbor	Displays details of established PIM neighbors.
show ip route detail	Displays the details of the unicast routing tables.
show mvpn bgp mdt-safi	Displays the BGP MDT SAFI database in MVPN.

Command	Purpose
show mvpn mdt encap	Displays the encapsulation table in MVPN. This table indicates how MVPN packets are encapsulated when sent out on the default vrf.
show mvpn mdt route	Displays details of the default and MDT routes. This data determines how customer data and control traffic is sent on the default VRF.
show routing [ip] multicast mdt encap	Displays the encapsulation table in the MRIB. This table indicates how MVPN packets are encapsulated when sent out on the default vrf.

Configuration Examples for MVPNs

This section includes the following configuration examples:

- Example: Configuring MVPN, page 32-29
- Example: Configuring the Multicast Address Range for Data MDTs, page 32-29

Example: Configuring MVPN

The following example shows how to configure an MVPN with two contexts:

```
vrf context vpn1
  ip pim rp-address 10.10.1.2 -list 224.0.0.0/8
  ip pim rp-address 10.10.1.3 -list 239.0.0.0/8 bidir
  ip pim ssm range 232.0.0.0/8
  mdt source loopback2
  mdt default 232.1.1.1
  mdt data 232.2.2.0/24 threshold 10 route-map rmap2
  mdt data bidir-enable
vrf context vpn4
  ip pim rp-address 10.10.4.2 -list 224.0.0.0/8
  ip pim rp-address 10.10.4.3 -list 239.0.0.0/8 bidir
  ip pim ssm range 232.0.0.0/8
  mdt default 235.1.1.1
  mdt asm-use-shared-tree
ip pim rp-address 10.11.0.2 -list 224.0.0.0/8
ip pim rp-address 10.11.0.3 -list 239.0.0.0/8 bidir
ip pim rp-address 10.11.0.4 -list 235.0.0.0/8
ip pim ssm range 232.0.0.0/8
```

Example: Configuring the Multicast Address Range for Data MDTs

The following example shows how to assign to the VPN routing instance a VRF named blue. The MDT default for a VPN VRF is 10.1.1.1, and the multicast address range for MDTs is 10.1.2.0 with wildcard bits of 0.0.0.3:

```
Vrf context blue
mdt data 239.1.0/24 threshold 10
```

Additional References for MVPNs

For additional information related to MVPN configuration, see the following sections:

- Related Documents
- Standards
- MIBs

Related Documents

Related Topic	Document Title
Multicast technology concepts	IP Multicast Technology Overview
VDCs	*Cisco Nexus 7000 Series NX-OS Virtual Device Context Configuration Guide, Release 5.x*
CLI commands	*Cisco Nexus 7000 Series NX-OS Multicast Routing Command Reference*
Basic IP multicast configuration	*Cisco Nexus 7000 Series NX-OS Multicast Routing Configuration Guide*

Standards

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

MIBs

MIB	MIBs Link
MPLS-VPN-MIB	To locate and download MIBs for selected platforms, releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for MVPNs

Table 32-2 lists the release history for this feature.

Table 32-2 Feature History for MVPNs

Feature Name	Releases	Feature Information
MVPNs	6.1(1)	Added support for multicast GRE tunnel interfaces for PE-CE routing with MVPN.
MVPNs	5.2(4)	Added support for multicast GRE tunnel interfaces for PE-CE routing with MVPN.
MVPN Intranet support	5.2(1)	This feature was introduced.

Configuring MPLS LSP Multipath Tree Trace

This chapter describes how to configure Multiprotocol Label Switching (MPLS) connectivity with the MPLS LSP Multipath Tree Trace feature.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LSP Multipath Tree Trace

.i.MPLS:tree trace;

The MPLS LSP Multipath Tree Trace feature provides the means to discover all possible equal-cost multipath (ECMP) routing paths of a label switched path (LSP) between an egress and ingress router. Once discovered, these paths can be retested on a periodic basis using MPLS LSP ping or traceroute. This feature is an extension to the MPLS LSP traceroute functionality for the tracing of IPv4 LSPs.

You can use the MPLS LSP Multipath Tree Trace feature to discover all paths for an IPv4 LSP.

This implementation of the MPLS LSP Multipath Tree Trace feature is based on the IETF RFC 4379 *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*.

This section includes the following topics:

Overview of MPLS LSP Multipath Tree Trace

As the number of MPLS deployments increases, the number of traffic types that the MPLS networks carry could increase. In addition, load balancing on label switch routers (LSRs) in the MPLS network provides alternate paths for carrying MPLS traffic to a target router. The ability of service providers to monitor LSPs and quickly isolate MPLS forwarding problems is critical to their ability to offer services.

Before the release of the MPLS LSP Multipath Tree Trace feature, no automated way existed to discover all paths between provider edge (PE) routers, and troubleshooting forwarding problems between PEs was difficult.

The MPLS LSP Multipath Tree Trace feature provides an automated way to discover all paths from the ingress PE router to the egress PE router in multivendor networks that use IPv4 load balancing at the transit routers. Once the PE-to-PE paths are discovered, use MPLS LSP ping and MPLS LSP traceroute to periodically test them.

Discovery of IPv4 Load Balancing Paths by MPLS LSP Multipath Tree Trace

.i.load balancing;

IPv4 load balancing at a transit router is based on the incoming label stack and the source and destination addresses in the IP header. The outgoing label stack and IP header source address remain constant for each branch being traced.

When you execute MPLS LSP multipath tree trace on the source LSR, the router needs to find the set of IP header destination addresses to use all possible output paths. The source LSR starts path discovery by sending a transit router a bitmap in an MPLS echo request. The transit router returns information in an MPLS echo request that contains subsets of the bitmap in a downstream map (DS Map) in an echo reply. The source router can then use the information in the echo reply to interrogate the next router. The source router interrogates each successive router until it finds one bitmap setting that is common to all routers along the path. The router uses TTL expiry to interrogate the routers to find the common bits.

For example, you could start path discovery by entering the following command at the source router:

```
switch# traceroute mpls multipath ipv4 10.131.101.129/32 hashkey ipv4 bitmap 16
```

This command sets the IP address of the target router as 10.131.101.192 255.255.255.255 and configures:

- The default hash key type to 8, which requests that an IPv4 address prefix and bit mask address set be returned in the DS Map in the echo reply.
- The bitmap size to 16. This means that MPLS LSP multipath tree trace uses 16 addresses (starting with 127.0.0.1) in the discovery of all paths of an LSP between the source router and the target router.

If you enter the **traceroute mpls multipath ipv4 10.131.101.129/32** command, MPLS LSP multipath tree trace uses the default hash type of 8 or IPv4 and a default bitmap size of 32. Your choice of a bitmap size depends on the number of routes in your network. If you have many routes, you might need to use a larger bitmap size.

Echo Reply Return Codes Sent by the Router Processing Multipath LSP Tree Trace

Table 33-1 describes the codes that the router processing a multipath LSP tree trace packet returns to the sender about the failure or success of the request.

Table 33-1 Echo Reply Return Codes

Output Code	Echo Return Code	Meaning
Period "."	—	A timeout occurred before the target router could reply.
x	0	No return code.
M	1	Malformed request.
m	2	Unsupported type, length, values (TLVs).
!	3	Success.
F	4	No Forwarding Equivalence Class (FEC) mapping.
D	5	DS Map mismatch.
R	6	Downstream router but not target.
U	7	Reserved.
L	8	Labeled output interface.
B	9	Unlabeled output interface.
f	10	FEC mismatch.
N	11	No label entry.
P	12	No receive interface label protocol.
p	13	Premature termination of the LSP.
X	unknown	Undefined return code.

Licensing Requirements for MPLS LSP Multipath Tree Trace

Product	License Requirement
Cisco NX-OS	The MPLS LSP Multipath Tree Trace feature requires an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LSP Multipath Tree Trace

The MPLS LSP Multipath Tree Trace feature has the following prerequisites:

- Before you can run MPLS ping and traceroute, ensure that the Intrusion Detection System (IDS) is disabled (specifically the option that drops packets if the IP address is in the reserved 127.x.x.x range).
- You must enable the MPLS LDP feature.

- You must understand the concepts and know how to use MPLS LSP ping or traceroute as described in the *MPLS LSP Ping/Traceroute for LDP/TE, and LSP Ping for VCCV* document.
- The routers in your network must use an implementation based on IETF RFC 4379 *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*.
- You should know the following about your MPLS network:
  - The topology
  - The number of links in your network
  - The expected number of LSPs, and how many LSPs
- Understand label switching, forwarding, and load balancing.

Guidelines and Limitations for MPLS LSP Multipath Tree Trace

The MPLS LSP Multipath Tree Trace feature has the following configuration guidelines and limitations:

- All restrictions that apply to the MPLS LSP ping and LSP traceroute features also apply to the MPLS LSP Multipath Tree Trace feature as follows:
  - You cannot use the MPLS LSP Multipath Tree Trace feature to trace the path taken by AToM packets. The MPLS LSP Multipath Tree Trace feature is not supported for AToM. (MPLS LSP ping is supported for AToM.) However, you can use the MPLS LSP Multipath Tree Trace feature to troubleshoot the Interior Gateway Protocol (IGP) LSP that is used by AToM.
  - You cannot use the MPLS LSP Multipath Tree Trace feature to validate or trace MPLS virtual private networks (VPNs). Multiple LSP paths are not discovered unless all routers in the MPLS core support an RFC 4379 implementation of *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures.*
- MPLS LSP multipath tree trace is not expected to operate in networks that support time-to-live (TTL) hiding.

Configuring MPLS LSP Multipath Tree Trace

This section includes the following topics:

- Requesting That a Transit Router Validate the Target FEC Stack for MPLS LSP Multipath Tree Trace, page 33-49
- Setting the Number of Timeout Attempts for MPLS LSP Multipath Tree Trace, page 33-50

Customizing the Default Behavior of MPLS Echo Packets

.i.customized echo packets;

You can customize the default behavior of MPLS echo packets. You might need to customize the default echo packet encoding and decoding behavior to allow later implementations of the *Detecting MPLS Data Plane Failures* (RFC 4379) to be deployed in networks running earlier versions of the draft.

MPLS Embedded Management Configuration

Before using the **ping mpls**, **traceroute mpls**, or **traceroute mpls multipath** command, you should ensure that the router is configured to encode and decode MPLS echo packets in a format that all receiving routers in the network can understand.

LSP ping drafts after Version 3 (draft-ietf-mpls-ping-03) have undergone numerous TLV format changes, but the implementations based on different drafts might not interoperate properly.

To allow later Cisco implementations to interoperate with draft Version 3 Cisco and non-Cisco implementations, a global configuration mode (MPLS OAM configuration) allows you to encode and decode echo packets in formats specified by draft Version 3 implementations.

Unless configured otherwise, a Cisco implementation encodes and decodes echo requests assuming the version on which the Internet Engineering Task Force (IETF) implementation is based.

To allow for seamless interoperability with earlier Revision 1 and 3 images, you can use MPLS Operation, Administration, and Maintenance (OAM) configuration mode parameters to force the default behavior of the Revision 4 images to be compliant or compatible in networks with Revision 1 or Revision 3 images.

To prevent failures reported by the replying router due to TLV version issues, you should configure all routers in the core. Encode and decode MPLS echo packets in the same draft version. For example, if the network is running RFC 4379 (Cisco Revision 4) implementations but one router can run only Version 3 (Cisco Revision 3), configure all routers in the network to operate in Revision 3 mode.

Cisco Revision 4 is the default version. The default version is the latest LSP ping version supported by the image on the router.

Prerequisites

The MPLS LSP Multipath Tree Trace feature requires RFC 4379 (Revision 4).

SUMMARY STEPS

1. **configure terminal**
2. **mpls oam**
3. **echo revision** {**3** | **4**}
4. [**no**] **echo vendor-extension**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal	Enters global configuration mode.
Step 2	**mpls oam** **Example:** switch(config)# mpls oam	Enters MPLS OAM configuration mode and customizes the default behavior of echo packets.
Step 3	**echo revision** {**3** | **4**} **Example:** switch(config-mpls)# echo revision 4	Customizes the default behavior of echo packets. • The **revision** keyword sets echo packet attributes to one of the following: – **3** = draft-ietf-mpls-ping-03 (Revision 2) – **4** = RFC 4379 compliant (default) **Note** The MPLS LSP Multipath Tree Trace feature requires Revision 4.
Step 4	[**no**] **echo vendor-extension** **Example:** switch(config-mpls)# echo vendor-extension	Customizes the default behavior of echo packets. • The **vendor-extension** keyword sends the Cisco-specific extension of TLVs with the echo packets. • The **no** form of the command allows you to disable a Cisco vendor's extension TLVs that another vendor's noncompliant implementations may not support. The router default is **echo vendor-extension**.

Configuring MPLS LSP Multipath Tree Trace

You can configure the MPLS multipath LSP tree trace traceroute. This task helps you to discover all LSPs from an egress router to an ingress router.

Prerequisites

Cisco LSP ping or traceroute implementations based on draft-ietf-mpls-lsp-ping-11 can in some cases detect the formatting of the sender of an MPLS echo request. However, in certain cases an echo request or echo reply might not contain the Cisco extension TLV. To avoid complications in which the echo packets are decoded assuming the wrong TLV formats, configure all routers in the network to operate in the same mode.

For an MPLS LSP multipath tree trace to be successful, the implementation in your routers must support RFC 4379 on all core routers.

If all routers in the network support FRC-4379 and another vendor's implementation exists that is not capable of properly handling Cisco's vendor TLV, the routers supporting the RFC-compliant or later configuration must include commands to disable the Cisco vendor TLV extensions.

SUMMARY STEPS

1. **configure terminal**
2. **mpls oam**
3. **echo revision 4**
4. (Optional) [**no**] **echo vendor-extension**
5. **traceroute mpls multipath ipv4** *destination-ip-address*/*destination-mask-length*

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**mpls oam** **Example:** `switch(config)# mpls oam`	Enters MPLS OAM configuration mode.
Step 3	**echo revision 4** **Example:** `switch(config-mpls)# echo revision 4`	Customizes the default behavior of echo packets. • The **revision 4** keywords set echo packet attributes to the default Revision 4 (RFC 4379 compliant). **Note** The MPLS LSP Multipath Tree Trace feature requires Revision 4.
Step 4	[**no**] **echo vendor-extension** **Example:** `switch(config-mpls) echo vendor-extension`	(Optional) Customizes the default behavior of echo packets. • The **vendor-extension** keyword sends the Cisco-specific extension of TLVs with the echo packets. • The **no** form of the command allows you to disable a Cisco vendor's extension TLVs that another vendor's noncompliant implementations may not support. The router default is **echo vendor-extension**.
Step 5	**traceroute mpls multipath ipv4** *destination-ip-address*/*destination-mask-length* **Example:** `switch# traceroute mpls multipath ipv4 10.131.161.251/32`	Discovers all LSPs from an egress router to an ingress router. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-ip-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required.

Discovering IPv4 Load Balancing Paths Using MPLS LSP Multipath Tree Trace

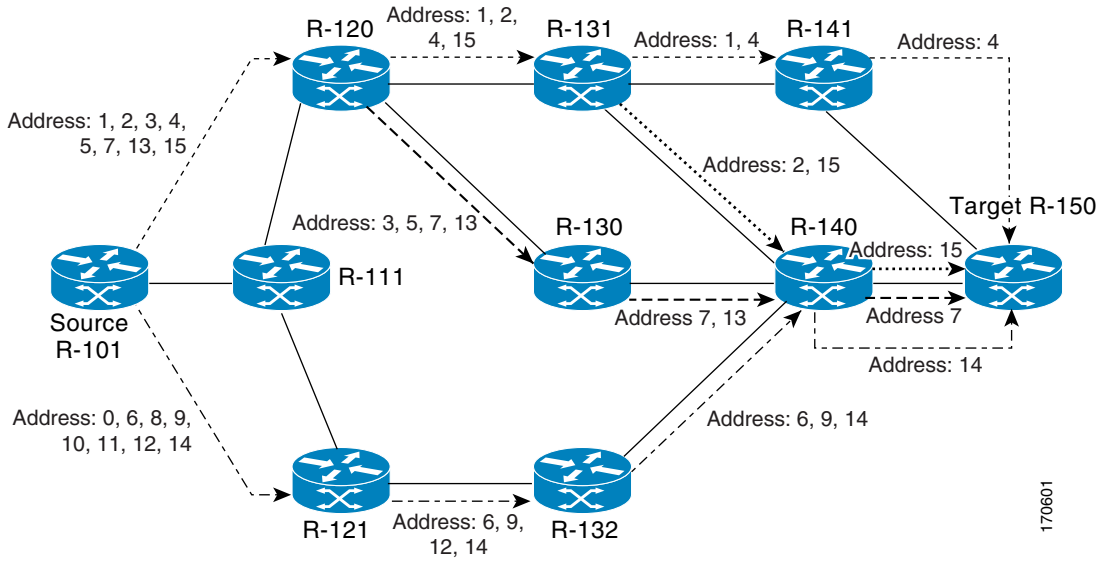
You can discover IPv4 load balancing paths using the MPLS LSP Multipath Tree Trace feature.

MPLS Multipath LSP Traceroute Path Discovery

A Cisco router load balances MPLS packets based on the incoming label stack and the source and destination addresses in the IP header. The outgoing label stack and IP header source address remain constant for each path being traced. The router needs to find the set of IP header destination addresses to use all possible output paths. This might require exhaustive searching of the 127.*x.y.z*/8 address space. Once you discover all paths from the source LSR to the target or destination LSR with the MPLS LSP Multipath Tree Trace feature, you can use MPLS LSP traceroute to monitor these paths.

Figure 33-1 shows how the MPLS LSP Multipath Tree Trace feature discovers LSP paths in a sample network. In Figure 33-1, the bitmap size is 16 and the numbers 0 to 15 represent the bitmapped addresses that the MPLS LSP Multipath Tree Trace feature uses to discover all the paths from the source LSR R-101 to the target LSR R-150. Figure 33-1 illustrates how the **traceroute mpls multipath** command discovers all LSP paths in the sample network.

Figure 33-1 MPLS LSP Multipath Tree Trace Path Discovery in a Sample Network



SUMMARY STEPS

1. **configure terminal**
2. **mpls oam**
3. **echo revision 4**
4. **traceroute mpls multipath ipv4** *destination-ip-address/destination-mask-length* **hashkey ipv4 bitmap** *bitmap-size*

DETAILED STEPS

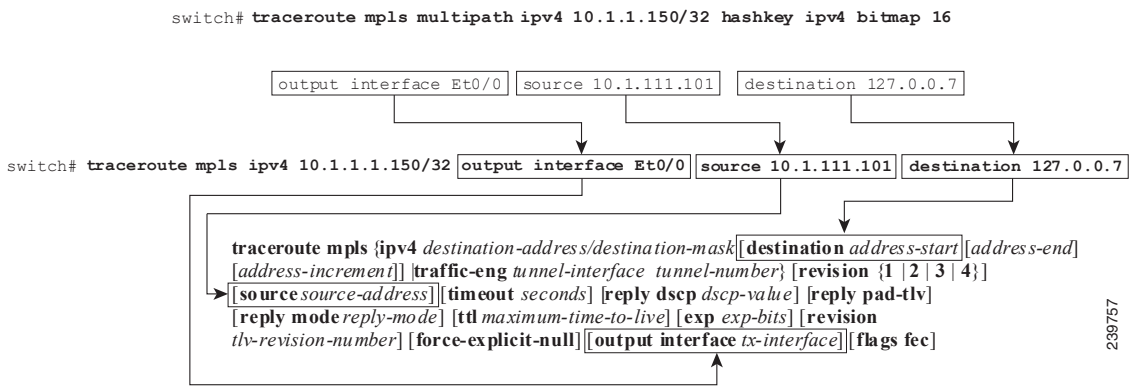
	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**mpls oam** **Example:** `switch(config)# mpls oam`	Enters MPLS OAM configuration mode and sets the echo packet attribute to Revision 4 (RFC 4379 compliant).
Step 3	**echo revision 4** **Example:** `switch(config-mpls)# echo revision 4`	Customizes the default behavior of echo packets. • The **revision 4** keywords set echo packet attributes to the default Revision 4 (RFC 4379 compliant). **Note** The MPLS LSP Multipath Tree Trace feature requires Revision 4.
Step 4	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* **hashkey ipv4 bitmap** *bitmap-size* **Example:** `switch# traceroute mpls multipath ipv4 10.131.161.251/32 hashkey ipv4 bitmap 16`	Discovers all MPLS LSPs from an egress router to an ingress router. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **hashkey ipv4** keywords set the hashkey type to IPv4 addresses. • The **bitmap** *bitmap-size* keyword and arguments set the bitmap size for multipath discovery.

Monitoring LSP Paths Discovered by MPLS LSP Multipath Tree Trace Using MPLS LSP Traceroute

You can monitor LSP paths that are discovered by the MPLS LSP Multipath Tree Trace feature using the MPLS LSP traceroute. You can take output directly from the **traceroute mpls multipath** command and add it to a **traceroute mpls** command periodically to verify that the path is still operating.

Figure 33-2 shows the mapping of the output of a **traceroute mpls multipath** command to a **traceroute mpls** command.

Figure 33-2 Mapping of traceroute mpls multipath Command Output to a traceroute mpls Command



Each path that you discover with the MPLS LSP Multipath Tree Trace feature can be tested in this manner periodically to monitor the LSP paths in your network.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address/destination-mask-length* **hashkey ipv4 bitmap** *bitmap-size*
2. **traceroute mpls ipv4** *destination-address/destination-mask-length* [**output interface** *tx-interface*] [**source** *source-address*] [**destination** *address-start*]

DETAILED STEPS

Step 1 Discover all MPLS LSPs from an egress router to an ingress router **by entering the traceroute mpls multipath ipv4** *destination-address/destination-mask-length* **hashkey ipv4 bitmap** *bitmap-size command.*

This example shows how to discover all MPLS LSPs from an egress router to an ingress router:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 hashkey ipv4 bitmap 16

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
```

```
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 468 ms
```

The output of the **traceroute mpls multipath** command in the example shows the result of path discovery with the MPLS LSP Multipath Tree Trace feature. In this example, the command sets the bitmap size to 16. Path discovery starts by the MPLS LSP Multipath Tree Trace feature using 16 bitmapped addresses as it locates LSP paths from the source router to the target router with prefix and mask 10.1.1.150/32. MPLS LSP multipath tree trace starts using the 127.*x.y.z*/8 address space with 127.0.0.1.

Step 2 Verify that the paths discovered when you entered a **traceroute mpls multipath** command are still operating by entering the **traceroute mpls ipv4** *destination-address/destination-mask-length* [**output interface** *tx-interface*] [**source** *source-address*] [**destination** *address-start*] command.

For example, the output for Path 0 in the previous **traceroute mpls multipath** command in Step 1 is as follows:

```
output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
```

If you put the output for path 0 in the **traceroute mpls** command, you see the following results:

```
switch# traceroute mpls ipv4 10.1.1.150/32 output interface Et0/0 source 10.1.111.101
destination 127.0.0.0

Tracing MPLS Label Switched Path to 10.1.1.150/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.1.111.101 MRU 1500 [Labels: 33 Exp: 0]
L 1 10.1.111.111 MRU 1500 [Labels: 34 Exp: 0] 40 ms
L 2 10.2.121.121 MRU 1500 [Labels: 34 Exp: 0] 32 ms
L 3 10.3.132.132 MRU 1500 [Labels: 32 Exp: 0] 16 ms
L 4 10.4.140.240 MRU 1504 [Labels: implicit-null Exp: 0] 20 ms
! 5 10.5.150.50 20 ms
```

You can take output directly from the **traceroute mpls multipath** command and add it to a **traceroute mpls** command periodically to verify that the path is still operating (see Figure 33-2).

Using DSCP to Request a Specific Class of Service in an Echo Reply

Use the reply differentiated services code point (DSCP) option to request a specific class of service (CoS) in an echo reply.

The reply DSCP option is supported in the experimental mode for IETF draft-ietf-mpls-lsp-ping-03.txt. Cisco implemented a vendor-specific extension for the reply DSCP option rather than using a Reply TOS TLV. A Reply TOS TLV serves the same purpose as the **reply dscp** command in IETF draft-ietf-mpls-lsp-ping-11.txt. This draft provides a standardized method of controlling the reply DSCP.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address/destination-mask-length* [**reply dscp** *dscp-value*]

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**reply dscp** *dscp-value*] **Example:** switch# traceroute mpls multipath ipv4 **10.131.191.252/32** reply dscp 50	Discovers all MPLS LSPs from an ingress router to an egress router and controls the DSCP value of an echo reply. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **reply dscp** *dscp-value* keywords and argument are the DSCP value of an echo reply. A Reply TOS TLV serves the same purpose as the **reply dscp** command in IETF draft-ietf-mpls-lsp-ping-11.txt. **Note** To specify a DSCP value, you must enter the **reply dscp** *dscp-value* keywords and argument.

Controlling How a Responding Router Replies to an MPLS Echo Request

This section describes how to control how a responding router replies to an MPLS echo request.

This section includes the following topic:

- Reply Modes for an MPLS LSP Multipath Tree Trace Echo Request Response, page 33-44

Reply Modes for an MPLS LSP Multipath Tree Trace Echo Request Response

The reply mode controls how a responding router replies to an MPLS echo request sent by a **traceroute mpls multipath** command. There are two reply modes for an echo request packet:

- ipv4—Reply with an IPv4 User Datagram Protocol (UDP) packet (default)
- router-alert—Reply with an IPv4 UDP packet with router alert

Note Use the ipv4 and router-alert reply modes with each other to prevent false negatives. If you do not receive a reply via the ipv4 mode, send a test with the router-alert reply mode. If both fail, something is wrong in the return path. The problem might be due to an incorrect ToS setting.

IPv4 UDP Reply Mode

The IPv4 UDP reply mode is the most common reply mode used with a **traceroute mpls multipath** command when you want to periodically poll the integrity of an LSP. With this option, you do not have explicit control over whether the packet traverses IP or MPLS hops to reach the originator of the MPLS echo request. If the originating (headend) router fails to receive a reply to an MPLS echo request when you use the **reply mode ipv4** keywords, use the **reply mode router-alert** keywords.

Router-Alert Reply Mode

The router-alert reply mode adds the router alert option to the IP header. When an IP packet that contains an IP router alert option in its IP header or an MPLS packet with a router alert label as its outermost label arrives at a router, the router punts (redirects) the packet to the supervisor process level for handling, which forces the supervisor of each intermediate router to handle the packet at each intermediate hop as it moves back to the destination. Hardware and line card forwarding inconsistencies are thus bypassed. Router-alert reply mode is slower than IPv4 mode because the reply requires process-level supervisor handling at each hop.

Table 33-2 describes how an incoming IP packet with an IP router alert is handled by the router switching path processes when the outgoing packet is an IP packet or an MPLS packet. It also describes how an MPLS packet with a router alert option is handled by the router switching path processes when the outgoing packet is an IP packet or an MPLS packet.

Table 33-2 Path Process Handling of IP and MPLS Router Alert Packets

Incoming Packet	Outgoing Packet	Software Switching Action
IP packet—Router alert option in IP header	IP packet—Router alert option in IP header	Forwards the packet as is.
	MPLS packet	Forwards the packet as is.
MPLS packet—Outermost label contains a router alert	IP packet—Router alert option in IP header	Removes the outermost router alert label and forwards the packet as an IP packet.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* **reply mode** {**ipv4** | **router-alert**}

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* **reply mode** {**ipv4** | **router-alert**} **Example:** switch# traceroute mpls multipath ipv4 **10.131.191.252/32** reply mode router-alert	Discovers all MPLS LSPs from an ingress router to an egress router and specifies the reply mode. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **reply mode** keyword requires that you enter one of the following keywords to specify the reply mode: – The **ipv4** keyword—Reply with an IPv4 UDP packet (default). – The **router-alert** keyword—Reply with an IPv4 UDP packet with router alert. **Note** To specify the reply mode, you must enter the **reply mode** keyword with the **ipv4** or **router-alert** keyword.

Specifying the Output Interface for Echo Packets Leaving a Router for MPLS LSP Multipath Tree Trace

You can specify the output interface for echo packets leaving a router for the MPLS LSP Multipath Tree Trace feature. You can use this task to test the LSPs that are reachable through a given interface.

You can control the interface through which packets leave a router. Path output information is used as input to LSP ping and traceroute.

The echo request output interface control feature allows you to force echo packets through the paths that perform detailed debugging or characterizing of the LSP. This feature is useful if a PE router connects to an MPLS cloud and there are broken links. You can direct traffic through a certain link. The feature also is helpful for troubleshooting network problems.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**output interface** *tx-interface*]

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**output interface** *tx-interface*] **Example:** `switch# traceroute mpls multipath ipv4 10.131.159.251/32 output interface ethernet0/0`	Discovers all MPLS LSPs from an ingress router to an egress router and specifies the interface through which echo packets leave a router. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **output interface** *tx-interface* keywords and argument specify the output interface for the MPLS echo request. **Note** You must specify the **output interface** keywords.

Setting the Pace of MPLS Echo Request Packet Transmission for MPLS LSP Multipath Tree Trace

You can set the pace of MPLS echo request packet transmission for the MPLS LSP Multipath Tree Trace feature. Echo request traffic pacing allows you to set the pace of the transmission of packets so that the receiving router does not drop packets. If you have a large amount of traffic on your network you might increase the size of the interval to help ensure that the receiving router does not drop packets.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**interval** *milliseconds*]

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**interval** *milliseconds*] **Example:** `switch# traceroute mpls multipath ipv4 10.131.159.251/32 interval 100`	Discovers all MPLS LSPs from an egress router to an ingress router and sets the time in milliseconds between successive MPLS echo requests. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **interval** *milliseconds* keyword and argument set the time between successive MPLS echo requests in milliseconds. The default is 0 milliseconds. **Note** To pace the transmission of packets, you must specify the **interval** keyword.

Enabling MPLS LSP Multipath Tree Trace to Detect LSP Breakages Caused by an Interface That Lacks an MPLS Configuration

You can enable the MPLS LSP Multipath Tree Trace feature to detect LSP breakages caused by an interface that lacks an MPLS configuration. If an interface is not configured for MPLS, then it cannot forward MPLS packets.

For an MPLS LSP Multipath Tree Trace of LSPs that carry IPv4 FECs, you can force an explicit null label to be added to the MPLS label stack even though the label was unsolicited. This process allows MPLS LSP multipath tree trace to detect LSP breakages that are caused by an interface that is not configured for MPLS. The MPLS LSP Multipath Tree Trace does not report that an LSP is functioning when it is unable to send MPLS traffic.

An explicit null label is added to an MPLS label stack if MPLS echo request packets are forwarded from an interface not configured for MPLS that is directly connected to the destination of the MPLS LSP Multipath Tree Trace or if the IP TTL value for the MPLS echo request packets is set to 1.

When you enter a **traceroute mpls multipath** command, you are looking for all MPLS LSP paths from an egress router to an ingress router. Failures at output interfaces that are not configured for MPLS at the penultimate hop are not detected. Explicit-null shimming allows you to test an LSP's ability to carry MPLS traffic.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address/destination-mask-length* **force-explicit-null**

DETAILED STEP

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* **force-explicit-null** **Example:** `switch# traceroute mpls multipath ipv4 10.131.191.252/32 force-explicit-null`	Discovers all MPLS LSPs from an egress router to an ingress router and forces an explicit null label to be added to the MPLS label stack. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **force-explicit-null** keyword forces an explicit null label to be added to the MPLS label stack even though the label was unsolicited. **Note** You must enter the **force-explicit-null** keyword to enable MPLS LSP multipath tree trace to detect LSP breakages caused by an interface that is not configured for MPLS.

Requesting That a Transit Router Validate the Target FEC Stack for MPLS LSP Multipath Tree Trace

You can request that a transit router validate the target Forwarding Equivalence Class (FEC) stack for the MPLS LSP Multipath Tree Trace feature.

An MPLS echo request tests a particular LSP. The LSP to be tested is identified by the FEC stack.

During an MPLS LSP Multipath Tree Trace, the echo packet validation rules do not require that a transit router validate the target FEC stack TLV. A downstream map TLV containing the correct received labels must be present in the echo request for target FEC stack checking to be performed.

To request that a transit router validate the target FEC stack, set the V flag from the source router by entering the **flags fec** keywords in the **traceroute mpls multipath** command. The default is that echo request packets are sent with the V flag set to 0.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**flags fec**] [**ttl** *maximum-time-to-live*]

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**flags fec**] [**ttl** *maximum-time-to-live*] **Example:** `switch# traceroute mpls multipath ipv4 10.131.159.252/32 flags fec ttl 5`	Discovers all MPLS LSPs from an egress router to an ingress router and requests validation of the target FEC stack by a transit router. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **flags fec** keywords request that the target FEC stack validation be done at a transit router. • The **ttl** *maximum-time-to-live* keyword and argument pair specify a maximum hop count. **Note** For a transit router to validate the target FEC stack, you must enter the **flags fec** and **ttl** keywords.

Setting the Number of Timeout Attempts for MPLS LSP Multipath Tree Trace

You can set the number of timeout attempts for the MPLS LSP Multipath Tree Trace feature.

A retry is tried if an outstanding echo request times out waiting for the corresponding echo reply.

SUMMARY STEPS

1. **traceroute mpls multipath ipv4** *destination-address/destination-mask-length* [**retry-count** *retry-count-value*]

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls multipath ipv4** *destination-address*/*destination-mask-length* [**retry-count** *retry-count-value*] **Example:** `switch# traceroute mpls multipath ipv4 10.131.159.252/32 retry-count 4`	Sets the number of retry attempts during an MPLS LSP multipath tree trace. • The **ipv4** keyword specifies the destination type as an LDP IPv4 address. • The *destination-address* argument is the address prefix of the target to be tested. • The *destination-mask-length* argument is the number of bits in the network mask of the target address. The **/** keyword before this argument is required. • The **retry-count** *retry-count-value* keyword and argument sets the number of retry attempts after a timeout occurs. A retry-count value of 0 means infinite retries. A retry-count value from 0 to10 is suggested. You might want to increase the retry value to greater than 10, if 10 is too small a value. The default retry-count value is 3. **Note** To set the number of retries after a timeout, you must enter the **retry-count** keyword.

Configuration Examples for MPLS LSP Multipath Tree Trace

This section includes the following configuration examples for the MPLS LSP Multipath Tree Trace feature:

- Example: Setting the Number of Timeout Attempts for MPLS LSP Multipath Tree Trace, page 33-59

Example: Customizing the Default Behavior of MPLS Echo Packets

The following example shows how to customize the behavior of MPLS echo packets so that the MPLS LSP Multipath Tree Trace feature interoperates with a vendor implementation that does not interpret RFC 4379 as Cisco does:

```
configure terminal
!
mpls oam
 echo revision 4
 no echo vendor-extension
```

The **echo revision** command is included in this example for completeness. The default echo revision number is 4, which corresponds to RFC 4379.

Example: Configuring MPLS LSP Multipath Tree Trace

The following example shows how to configure the MPLS LSP Multipath Tree Trace feature to interoperate with a vendor implementation that does not interpret RFC 4379 as Cisco does:

```
configure terminal
!
mpls oam
 echo revision 4
 no echo vendor-extension
!
traceroute mpls multipath ipv4 10.131.161.151/32
```

The **echo revision** command is included in this example for completeness. The default echo revision number is 4, which corresponds to the RFC 4379.

Example: Discovering IPv4 Load Balancing Paths Using MPLS LSP Multipath Tree Trace

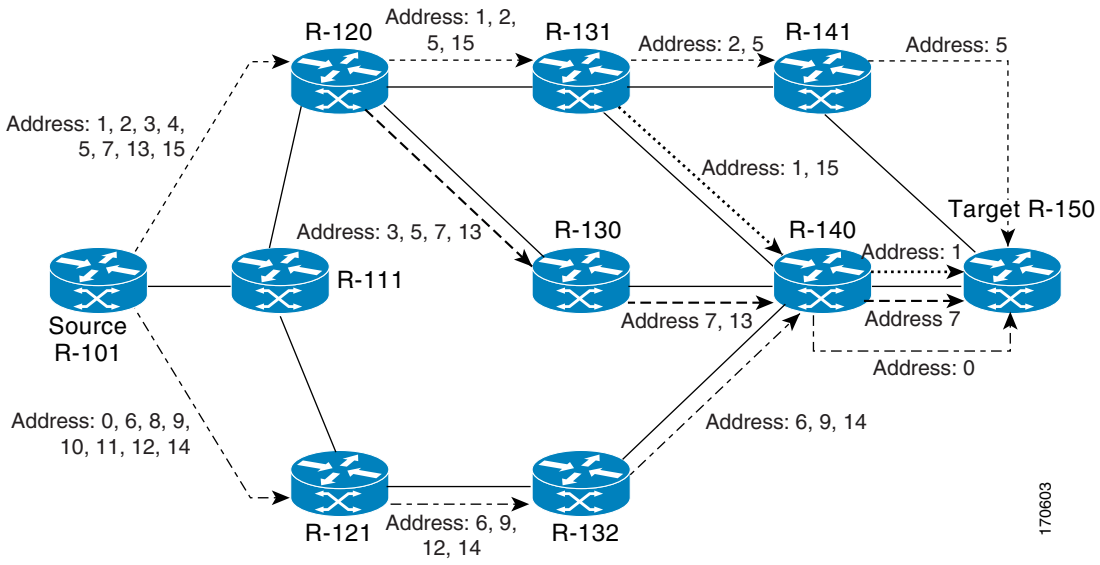
The following example shows how to use the MPLS LSP Multipath Tree Trace feature to discover IPv4 load-balancing paths. The example is based on the sample network shown in Figure 33-3. In this example, the bitmap size is set to 16. Therefore, path discovery starts by the MPLS LSP Multipath Tree Trace feature using 16 bitmapped addresses as it locates LSP paths from the source router R-101 to the target router R-150 with prefix and mask 10.1.1.150/32. The MPLS LSP Multipath Tree Trace feature starts using the 127.*x.y.z*/8 address space with 127.0.0.0.

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 hashkey ipv4 bitmap 16

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0
```

```
Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 468 ms
```

The output of the **traceroute mpls multipath** command in the example shows the result of path discovery with the MPLS LSP Multipath Tree Trace feature as shown in Figure 33-3.

Figure 33-3 MPLS LSP Multipath Tree Trace Path Discovery in a Sample Network



Example: Using DSCP to Request a Specific Class of Service in an Echo Reply

The following example shows how to use DSCP to request a specific Class of Service (CoS) in an echo reply:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 reply dscp 50

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
```

```
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 448 ms
```

Example: Controlling How a Responding Router Replies to an MPLS Echo Request

The following example shows how to control how a responding router replies to an MPLS echo request:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 reply mode router-alert

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0
 Total Time Elapsed 708 ms
```

Example: Specifying the Output Interface for Echo Packets Leaving a Router for MPLS LSP Multipath Tree Trace

The following example shows how to specify the output interface for echo packets leaving a router for the MPLS LSP Multipath Tree Trace feature:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 output interface ethernet0/0

Tracing MPLS Label Switched Path to 10.1.1.150/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.1.111.101 MRU 1500 [Labels: 33 Exp: 0]
L
  1 10.1.111.111 MRU 1500 [Labels: 33 Exp: 0] 40 ms
L
  2 10.2.120.120 MRU 1500 [Labels: 33 Exp: 0] 20 ms
L
  3 10.3.131.131 MRU 1500 [Labels: 34 Exp: 0] 20 ms
L
  4 10.4.141.141 MRU 1504 [Labels: implicit-null Exp: 0] 20 ms !
  5 10.5.150.150 16 ms
```

Example: Setting the Pace of MPLS Echo Request Packet Transmission for MPLS LSP Multipath Tree Trace

The following examples show how set the pace of MPLS echo request packet transmission for the MPLS LSP Multipath Tree Trace feature. The time between successive MPLS echo requests is set to 300 milliseconds in the first example and 400 milliseconds in the second example:

```
switch# traceroute mpls multipath ipv4 10.131.159.252/32 interval 300

Starting LSP Multipath Traceroute for 10.131.159.252/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LL!
Path 0 found,
 output interface Et1/0 source 10.2.3.2 destination 127.0.0.0

Paths (found/broken/unexplored) (1/0/0)
 Echo Request (sent/fail) (3/0)
 Echo Reply (received/timeout) (3/0)
 Total Time Elapsed 1604 ms
```

```
switch# traceroute mpls multipath ipv4 10.131.159.252/32 interval 400

Starting LSP Multipath Traceroute for 10.131.159.252/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LL!
Path 0 found,
 output interface Et1/0 source 10.2.3.2 destination 127.0.0.0

Paths (found/broken/unexplored) (1/0/0)
 Echo Request (sent/fail) (3/0)
 Echo Reply (received/timeout) (3/0)
 Total Time Elapsed 1856 ms
```

Notice that the elapsed time increases as you increase the interval size.

Example: Enabling MPLS LSP Multipath Tree Trace to Detect LSP Breakages Caused by an Interface That Lacks an MPLS Configuration

The following examples shows how to enable the MPLS LSP Multipath Tree Trace feature to detect LSP breakages caused by an interface that lacks an MPLS configuration:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 force-explicit-null

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 460 ms
```

This example shows the additional information provided when you add the **verbose** keyword to the command:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 force-explicit-null verbose

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
 'L' - labeled output interface, 'B' - unlabeled output interface,
 'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
 'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
 'P' - no rx intf label prot, 'p' - premature termination of LSP,
 'R' - transit router, 'I' - unknown upstream index,
 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
  0 10.1.111.101 10.1.111.111 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] multipaths 0
L
  1 10.1.111.111 10.2.121.121 MRU 1500 [Labels: 34/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
  2 10.2.121.121 10.3.132.132 MRU 1500 [Labels: 34/explicit-null Exp: 0/0] ret code 8
multipaths 1
L
  3 10.3.132.132 10.4.140.240 MRU 1500 [Labels: 32/explicit-null Exp: 0/0] ret code 8
multipaths 1
L
  4 10.4.140.240 10.5.150.50 MRU 1504 [Labels: explicit-null Exp: 0] ret code 8 multipaths
1 !
  5 10.5.150.50, ret code 3 multipaths 0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
  0 10.1.111.101 10.1.111.111 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] multipaths 0
L
  1 10.1.111.111 10.2.120.120 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
  2 10.2.120.120 10.3.131.131 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
  3 10.3.131.131 10.4.141.141 MRU 1500 [Labels: 34/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
  4 10.4.141.141 10.5.150.150 MRU 1504 [Labels: explicit-null Exp: 0] ret code 8
multipaths 1
!
5 10.5.150.150, ret code 3 multipaths 0
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
  0 10.1.111.101 10.1.111.111 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] multipaths 0
L
 1 10.1.111.111 10.2.120.120 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
 2 10.2.120.120 10.3.131.131 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
 3 10.3.131.131 10.4.140.140 MRU 1500 [Labels: 32/explicit-null Exp: 0/0] ret code 8
multipaths 2
```

```
L
 4 10.4.140.140 10.5.150.50 MRU 1504 [Labels: explicit-null Exp: 0] ret code 8 multipaths
1 ! 5 10.5.150.50, ret code 3 multipaths 0
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7
  0 10.1.111.101 10.1.111.111 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] multipaths 0
L
  1 10.1.111.111 10.2.120.120 MRU 1500 [Labels: 33/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
  2 10.2.120.120 10.3.130.130 MRU 1500 [Labels: 34/explicit-null Exp: 0/0] ret code 8
multipaths 2
L
  3 10.3.130.130 10.4.140.40 MRU 1500 [Labels: 32/explicit-null Exp: 0/0] ret code 8
multipaths 1
L
  4 10.4.140.40 10.5.150.50 MRU 1504 [Labels: explicit-null Exp: 0] ret code 8 multipaths
1
!
  5 10.5.150.50, ret code 3 multipaths 0

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 492 ms
```

Example: Requesting That a Transit Router Validate the Target FEC Stack for MPLS LSP Multipath Tree Trace

The following example shows how to request that a transit router validate the target FEC stack for the MPLS LSP Multipath Tree Trace feature:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 flags fec ttl 5

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
```

```
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 464 ms
```

Target FEC stack validation is always done at the egress router when the **flags fec** keywords are specified in the **traceroute mpls multipath** command.

Example: Setting the Number of Timeout Attempts for MPLS LSP Multipath Tree Trace

The following example sets the number of timeout attempts for the MPLS LSP Multipath Tree Trace feature to four:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 retry-count 4

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
LLLL!
Path 0 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1
L!
Path 2 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.5
LL!
Path 3 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (4/0/0)
 Echo Request (sent/fail) (14/0)
 Echo Reply (received/timeout) (14/0)
 Total Time Elapsed 460 ms
```

The following output shows a **traceroute mpls multipath** command that found one unexplored path, one successful path, and one broken path:

```
switch# traceroute mpls multipath ipv4 10.1.1.150/32 retry-count 4

Starting LSP Multipath Traceroute for 10.1.1.150/32

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
```

```
LLL....
Path 0 Unexplorable,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.0
LLL!
Path 1 found,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.1 B
Path 2 Broken,
 output interface Et0/0 source 10.1.111.101 destination 127.0.0.7

Paths (found/broken/unexplored) (1/1/1)
 Echo Request (sent/fail) (12/0)
 Echo Reply (received/timeout) (8/4)
 Total Time Elapsed 7868 ms
```

Additional References for MPLS LSP Multipath Tree Trace

For additional information related to the MPLS LSP Multipath Tree Trace feature, see the following sections:

- Related Documents, page 33-60
- MIBs, page 33-60

Related Documents

Related Topic	Document Title
Cisco NX-OS MPLS commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Feature History for MPLS LSP Multipath Tree Trace

Table 33-3 lists the release history for this feature.

Table 33-3 Feature History for MPLS LSP Multipath Tree Trace

Feature Name	Releases	Feature Information
MPLS LSP multipath tree trace	5.2(1)	This feature was introduced.

Verifying Connectivity with MPLS LSP Ping and Traceroute

This chapter describes how to verify Multiprotocol Label Switching (MPLS) connectivity with the MPLS label switched protocol (LSP) ping and traceroute feature.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "New and Changed Information" chapter or the Feature History table below.

Information About MPLS LSP Ping and Traceroute

.i.MPLS:ping;
.i.MPLS:traceroute;
MPLS LSP ping and traceroute helps operators to monitor label switched paths (LSPs) and quickly isolate MPLS forwarding problems. You use MPLS LSP ping and traceroute to test LSP connectivity for IPv4 Label Distribution Protocol (LDP) prefixes and Resource Reservation Protocol (RSVP) traffic engineering (TE) LSPs.

Internet Control Message Protocol (ICMP) ping and traceroute are used to help diagnose the root cause when a forwarding failure occurs. However, ping and traceroute might not detect LSP failures because an ICMP packet can be forwarded through IP to the destination when an LSP breakage occurs.

MPLS LSP ping and traceroute are well suited for identifying LSP breakages for the following reasons:

- An MPLS echo request packet cannot be forwarded through IP because its IP Time to Live (TTL) is set to 1 and its IP destination address field is set to a 127/8 address.
- The Forwarding Equivalence Class (FEC) being checked is not stored in the IP destination address field (as is the case of ICMP).

MPLS echo request and reply packets test LSPs. The features described in this chapter are based on the IETF RFC 4379 *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*:

.i.MPLS:echo request packets;
.i.MPLS:echo reply packets;

- Echo request output interface control
- Echo request traffic pacing
- Echo request end-of-stack explicit-null label shimming
- Echo request request-dsmap capability
- Request-fec checking
- Depth limit reporting

The section includes the following topics:

MPLS LSP Ping Operation

You can use MPLS LSP echo request and reply packets to validate an LSP by using the **ping mpls** command.

The MPLS echo request packet is sent to a target router through the use of the appropriate label stack associated with the LSP to be validated. Use of the label stack causes the packet to be forwarded over the LSP.

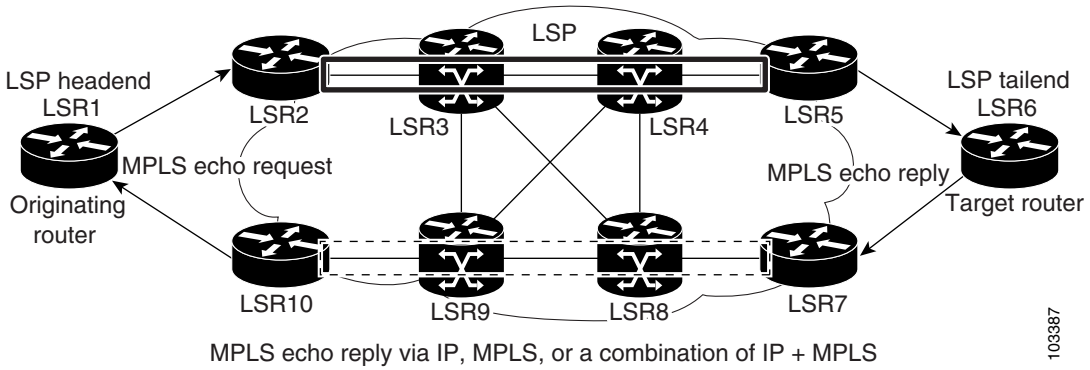
The destination IP address of the MPLS echo request packet is different from the address used to select the label stack. The destination IP address is defined as a 127.*x*.*y*.*z*/8 address. The 127.*x*.*y*.*z*/8 address prevents the IP packet from being forwarded over IP to its destination if the LSP is broken.

An MPLS echo reply is sent in response to an MPLS echo request. The reply is sent as an IP packet and it is forwarded using IP, MPLS, or a combination of both types of switching. The source address of the MPLS echo reply packet is an address obtained from the router that is generating the echo reply. The destination address is the source address of the router that originated the MPLS echo request packet.

The MPLS echo reply destination port is set to the echo request source port.

Figure 34-1 shows MPLS LSP ping echo request and echo reply paths.

Figure 34-1 MPLS LSP Ping Echo Request and Echo Reply Paths



If you initiate an MPLS LSP ping request at LSR1 to a FEC at LSR6, you get the results shown in Table 34-1.

Table 34-1 MPLS LSP Ping Example from the Preceding Figure

Step	Router	Action
1.	LSR1	Initiates an LSP ping request for an FEC at the target router LSR6 and sends an MPLS echo request to LSR2.
2.	LSR2	Receives the MPLS echo request packet and forwards it through transit routers LSR3 and LSR4 to the penultimate router LSR5.
3.	LSR5	Receives the MPLS echo request, pops the MPLS label, and forwards the packet to LSR6 as an IP packet.
4.	LSR6	Receives the IP packet, processes the MPLS echo request, and sends an MPLS echo reply to LSR1 through an alternate route.
5.	LSR7 to LSR10	Receives the MPLS echo reply and forwards it back toward LSR1, the originating router.
6.	LSR1	Receives the MPLS echo reply in response to its MPLS echo request.

Ping Draft Versions

LSP ping drafts after Version 3 (draft-ietf-mpls-ping-03) have undergone numerous TLV format changes, but the versions of the draft do not always interoperate.

Unless configured otherwise, a Cisco implementation encodes and decodes echo requests assuming the version on which the IETF implementations is based.

To prevent failures reported by the replying device due to TLV version issues, you should configure all devices in the core. Encode and decode MPLS echo packets in the same draft version.

Cisco Vendor Extensions

In Cisco's Version 3 (draft-ietf-mpls-ping-03.txt) implementations, Cisco defined a vendor extension type, length, value (TLV) in the ignore-if-not-understood TLV space. It is used to provide the following capabilities:

- Provides an ability to track TVL versions—This capability was defined before the existence of the global configuration command for setting the echo packet encode and decode behavior. The TLV version information in an echo packet overrides the configured decoding behavior. Using this TLV for TLV versions is no longer required since the introduction of the global configuration capability.
- Provides an experimental reply Type of Service (ToS)—This capability controls the reply differentiated services code point (DSCP). Because Draft Version 8 defines a reply ToS TLV, the use of the reply DSCP is no longer required.

You enable compatibility between the MPLS LSP and ping or traceroute implementation by customizing the default behavior of echo packets.

MPLS LSP Traceroute Operation

.i.traceroute mpls command;
.i.MPLS:traceroute process;

MPLS LSP traceroute uses MPLS echo request and reply packets to validate an LSP. You can use MPLS LSP traceroute to validate IPv4 LDP and IPv4 RSVP FECs by using appropriate keywords and arguments with the **traceroute mpls** command.

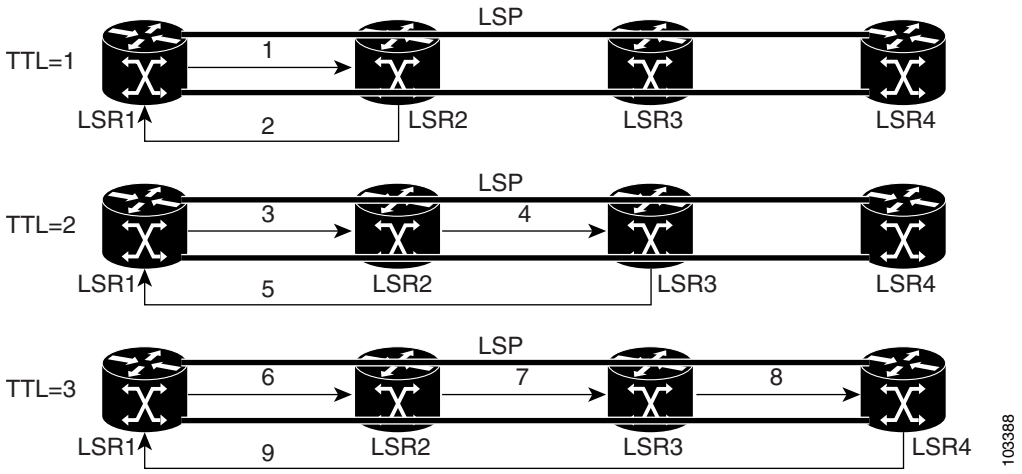
MPLS LSP traceroute uses Time-to-Live (TTL) settings to force TTL along an LSP to expire. MPLS LSP traceroute incrementally increases the TTL value in its MPLS echo requests (TTL = 1, 2, 3, 4) to discover the downstream mapping of each successive hop. The transit router processes the MPLS echo request when it receives a labeled packet with a TTL = 1. When the TTL expires, the transit router sends the packet to the supervisor for processing and the transit router returns an MPLS echo reply that contains information about the transit hop in response to the TTL-expired MPLS packet.

The MPLS echo reply destination port is set to the echo request source port.

Note When a router traces an IPV4 FEC that goes over a traffic engineering (TE) tunnel, intermediate routers might return U (unreachable) if LDP is not running in those intermediate routers.

The following figure shows an MPLS LSP traceroute example with an LSP from LSR1 to LSR4.

Figure 34-2 MPLS LSP Traceroute Example



If you enter an MPLS LSP traceroute to an FEC at LSR4 from LSR1, you get the results shown in Table 34-2.

Table 34-2 MPLS LSP Traceroute Example Based on the Preceding Figure

Step	Router	MPLS Packet Type and Description	Router Action (Receive or Send)
1.	LSR1	MPLS echo request—With a target FEC pointing to LSR4 and to a downstream mapping	• Sets the TTL of the label stack to 1 • Sends the request to LSR2
2.	LSR2	MPLS echo reply	• Receives the packet with a TTL = 1 • Processes the User Datagram Protocol (UDP) packet as an MPLS echo request • Finds a downstream mapping and replies to LSR1 with its own downstream mapping, based on the incoming label
3.	LSR1	MPLS echo request—With the same target FEC and the downstream mapping received in the echo reply from LSR2	• Sets the TTL of the label stack to 2 • Sends the request to LSR2
4.	LSR2	MPLS echo request	• Receives the packet with a TTL = 2 • Decrements the TTL • Forwards the echo request to LSR3
5.	LSR3	MPLS reply packet	• Receives the packet with a TTL = 1 • Processes the UDP packet as an MPLS echo request • Finds a downstream mapping and replies to LSR1 with its own downstream mapping based on the incoming label
6.	LSR1	MPLS echo request—With the same target FEC and the downstream mapping received in the echo reply from LSR3	• Sets the TTL of the packet to 3 • Sends the request to LSR2

Table 34-2 MPLS LSP Traceroute Example Based on the Preceding Figure (continued)

Step	Router	MPLS Packet Type and Description	Router Action (Receive or Send)
7.	LSR2	MPLS echo request	• Receives the packet with a TTL = 3 • Decrements the TTL • Forwards the echo request to LSR3
8.	LSR3	MPLS echo request	• Receives the packet with a TTL = 2 • Decrements the TTL • Forwards the echo request to LSR4
9.	LSR4	MPLS echo reply	• Receives the packet with a TTL = 1 • Processes the UDP packet as an MPLS echo request • Finds a downstream mapping and also finds that the router is the egress router for the target FEC • Replies to LSR1

MPLS Network Management with MPLS LSP Ping and MPLS LSP Traceroute

To manage an MPLS network, you must be able to monitor LSPs and quickly isolate MPLS forwarding problems. You need ways to characterize the liveliness of an LSP and reliably detect when an LSP fails to deliver user traffic.

You can use MPLS LSP ping to verify the LSP that is used to transport packets destined for IPv4 LDP prefixes. You can use MPLS LSP traceroute to trace LSPs that are used to carry packets destined for IPv4 LDP prefixes.

An MPLS echo request is sent through an LSP to validate it. A TTL expiration or LSP breakage causes the transit router to process the echo request before it gets to the intended destination. The router returns an MPLS echo reply that contains an explanatory reply code to the originator of the echo request.

The successful echo request is processed at the egress of the LSP. The echo reply is sent through an IP path, an MPLS path, or a combination of both back to the originator of the echo request.

Information Provided by the Router Processing LSP Ping or LSP Traceroute

Table 34-3 describes the codes that the router processing an LSP ping or LSP traceroute packet returns to the sender about the failure or success of the request.

You can also display the return code for an MPLS LSP ping operation if you enter the **verbose** keyword with the **ping mpls** command.

Table 34-3 Echo Reply Return Codes

Output Code	Echo Return Code	Meaning
x	0	No return code.
M	1	Malformed echo request.
m	2	Unsupported TLVs.
!	3	Success.

Table 34-3 Echo Reply Return Codes (continued)

Output Code	Echo Return Code	Meaning
F	4	No FEC mapping.
D	5	DS map mismatch.
I	6	Unknown upstream interface index.
U	7	Reserved.
L	8	Labeled output interface.
B	9	Unlabeled output interface.
f	10	FEC mismatch.
N	11	No label entry.
P	12	No receive interface label protocol.
p	13	Premature termination of the LSP.
X	unknown	Undefined return code.

Note Echo return codes 6 and 7 are accepted only for Version 3 (draft-ieft-mpls-ping-03). For version_3, these return codes have the following meaning:

- Code 6: The replying router is one of the downstream routers and its mapping for this FEC on the received interface is the given label.
- Code 7: The replying router is one of the downstream routers, but its mapping for this FEC is not the given label.

IP Does Not Forward MPLS Echo Request Packets

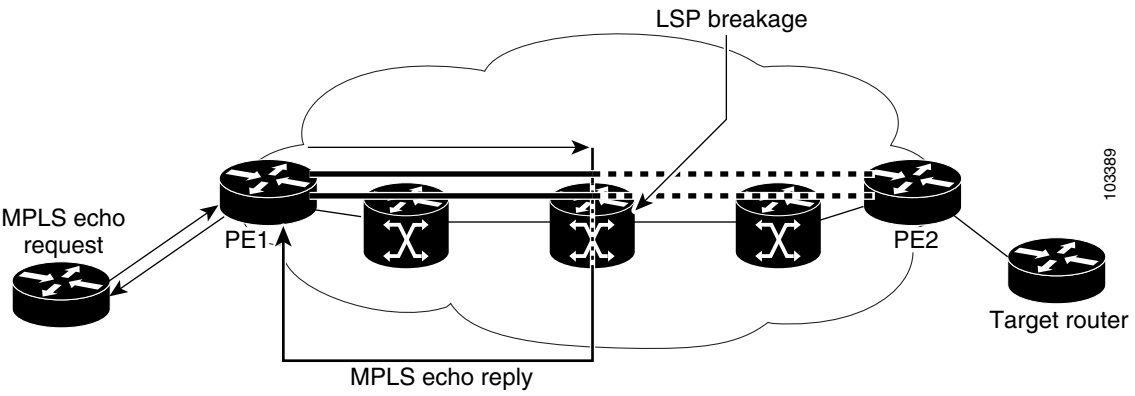
MPLS echo request packets that are sent during an LSP ping are never forwarded by IP. The IP header destination address field in an MPLS echo request packet is a 127.*x.y.z*/8 address. Routers should not forward packets using a 127.*x.y.z*/8 address. The 127.*x.y.z*/8 address corresponds to an address for the local host.

Using a 127.*x.y.z* address as the destination address of the UDP packet is significant because the MPLS echo request packet might fail to make it to the target router if the transit router does not label switch the LSP. Using a 127.*x.y.z* address allows for the detection of LSP breakages. The following occurs at the transit router:

- If an LSP breakage occurs at a transit router, the MPLS echo packet is not forwarded; it is consumed by the router.
- If the LSP is intact, the MPLS echo packet reaches the target router and is processed by the terminal point of the LSP.

The following figure shows the path of the MPLS echo request and reply when a transit router fails to label switch a packet in an LSP.

Figure 34-3 Path of a Transit Router that Fails to Label Switch a Packet



Note An MPLS virtual private network (VPN) packet, although an IP packet, does not contain usable forwarding information at a transit router because the destination IP address is significant only to the virtual routing and forwarding (VRF) instances at the endpoints of the MPLS network.

Virtual Circuit Connectivity Verification

Virtual Circuit Connectivity Verification (VCCV) in Layer 2 VPN Operations, Administration, and Maintenance (OAM) is used for fault detection and diagnostic of the pseudowire (PW). VCCV defines a set of messages that are exchanged on the PW to verify the connectivity. VCCV messages should be encapsulated so that the messages traverse the same path as the normal data in a network. VCCV also defines the use of the out-of-band to send VCCV messages. The Control Channel (CC) types defined for VCCV are as follows:

- Type 1—Uses a control word with 0001b as the first nibble. The VCCV packets traverse the same path as the data on the network. Type 1 cannot be used when the control word is not used in the pseudowire.
- Type 2—Uses a Router Alert Label in the encapsulation. Because a new label is added between the VC label and tunnel label(s), the packets might not be able to follow the same path as the data because of the equal cost multipath (ECMP) hashing in the core routers. Type 2 is the preferred method when the control word is not used in the pseudowire.
- Type 3—Uses Time To Live (TTL) in the Virtual Circuit (VC) label. Type 3 is used when the control word is used in the pseudowire.

Once the VCCV packet is delivered to the endpoint of the pseudowire, the Connectivity Verification (CV) types are used to determine the type of VCCV message. The following types of messages are defined for VCCV:

- Internet Control Message Protocol (ICMP) ping
- Label switch path (LSP) ping
- Bidirectional Forwarding Detection (BFD) for pseudowire fault detection
- BFD for pseudowire fault detection and status signaling
- BFD for pseudowire fault detection only without an IP header
- BFD for pseudowire fault detection and status signaling without an IP header

VCCV defines a set of messages that are exchanged between PEs to verify connectivity of the pseudowire. To make sure that pseudowire packets follow the same path as the data flow, they are encapsulated with the same labels. You can use VCCV as a diagnostic tool or as a fault detection tool.

In the diagnostic mode, you can trigger the LSP ping or ICMP ping modes depending on the underlying Public Services Network (PSN). Because a pseudowire is bidirectional, you should require that the reply be sent over the PSN tunnel that makes up the other half of the PW under test. For example, if the PSN is an MPLS LSP, send the reply on the LSP that represents the reverse path. If this process fails, you can use other reply modes to determine what is wrong.

The fault detection mode enables you to emulate a fault for detection mechanisms in other technologies, such as asynchronous transfer mode (ATM). In the fault detection mode, the upstream provider edge (PE) sends BFD control messages periodically. When the downstream PE does not receive these messages for a defined period of time, it declares that direction of the PW as down and notifies the upstream PE. Based on the emulated service, the PEs may send indications over the related attachment circuits to notify the end points of the fault condition.

Licensing Requirements for MPLS LSP Ping and Traceroute

Product	License Requirement
Cisco NX-OS	MPLS LSP ping and traceroute require an MPLS license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for MPLS LSP Ping and Traceroute

MPLS LSP ping and traceroute have the following prerequisites:

- You must install the MPLS license.
- Before you can run MPLS LSP ping and traceroute, ensure that the Intrusion Detection System (IDS) is disabled, specifically the option that drops packets if the IP address is in the reserved 127.x.x.x range.
- You must enable the MPLS LDP feature or the MPLS traffic engineering (TE) feature.

Guidelines and Limitations for MPLS LSP Ping and Traceroute

MPLS LSP ping and traceroute have the following configuration guidelines and limitations:

- You cannot use MPLS LSP ping to validate or trace MPLS VPNs.
- You cannot use MPLS LSP traceroute to troubleshoot LSPs that use TTL hiding.
- MPLS supports per-destination and per-packet (round robin) load balancing. If per-packet load balancing is in effect, you should not use MPLS LSP traceroute because during an LSP traceroute, a transit router makes consistency checks on the information supplied in the previous echo response from the directly connected upstream router. When you use round robin, you cannot control the path that an echo request packet takes in a way that allows a packet to be directed to TTL expire at a given router. Without that ability, the consistency check might fail during an LSP traceroute, and a consistency check failure return code might appear.

- A platform must support LSP ping and traceroute in order to respond to an MPLS echo request packet.
- Unless you enable the MPLS LSP Ping/Traceroute for LDP/TE , and LSP Ping for the virtual circuit connection verification (VCCV) feature along the entire path, you cannot get a reply if the request fails along the path at any node.
- The draft version on other devices in the network must be compatible with the draft version implemented on Cisco NX-OS. Earlier versions might not be compatible with later versions because of changes to type, length, and values (TLVs) without sufficient versioning information.
- You cannot use MPLS LSP traceroute to trace the path taken by Any Transport over MPLS (AToM) packets. However, you can use MPLS LSP traceroute to troubleshoot the Interior Gateway Protocol (IGP) LSP that is used by AToM.
- You cannot use MPLS LSP traceroute to troubleshoot LSPs that employ Time-to-Live (TTL) hiding. If you want to use MPLS LSP traceroute, the network should not use TTL hiding.

Configuring MPLS LSP Ping and Traceroute

This section includes the following topics:

Enabling Compatibility Between the MPLS LSP and Ping or Traceroute Implementation

LSP ping drafts after Version 3 (draft-ietf-mpls-ping-03) have undergone many TLV format changes, but the versions of the draft do not always interoperate.

Unless configured otherwise, a Cisco implementation encodes and decodes echo requests by assuming the version on which the IETF implementations are based.

To ensure interoperability among devices and prevent failures reported by the replying router due to TLV version issues, you should configure all routers in the core to encode and decode MPLS echo packets in the same draft version. For example, if the network is running RFC 4379 (Cisco Version 4) implementations but one router is capable of only Version 3 (Cisco Revision 3), configure all routers in the network to operate in Revision 3 mode.

Cisco Vendor Extensions

In Cisco's Version 3 (draft-ietf-mpls-ping-03.txt) implementations, Cisco defined a vendor extension TLV in the ignore-if-not-understood TLV space. It is used for the following purposes:

- Provide an ability to track TLV versions.

- Provide an experimental Reply type of service (ToS) capability.

The first capability was defined before the existence of the global configuration command for setting the echo packet encode and decode behavior. TLV version information in an echo packet overrides the configured decoding behavior. Using this TLV for TLV versions is no longer required since the introduction of the global configuration capability.

The second capability controls the reply DSCP. Draft Version 8 defines a Reply ToS TLV, so the use of the reply DSCP is no longer required.

SUMMARY STEPS

1. **configure terminal**
2. **mpls oam**
3. **echo revision** {**3** | **4**}
4. **echo vendor-extension**
5. **exit**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**mpls oam** **Example:** `switch(config)# mpls oam`	Enters MPLS OAM configuration mode for customizing the default behavior of echo packets.
Step 3	**echo revision** {**3** | **4**} **Example:** `switch(config-mpls)# echo revision 4`	Specifies the revision number of the echo packet's default values. • 3—draft-ietf-mpls-ping-03 (Revision 2). • 4—RFC 4379 compliant (default).
Step 4	**echo vendor-extension** **Example:** `switch(config-mpls)# echo vendor-extension`	Sends the Cisco-specific extension of TLVs with echo packets.
Step 5	**exit** **Example:** `switch(config-mpls)# exit`	Returns to global configuration mode.

Validating an LDP IPv4 FEC

An LSP is formed by labels. Routers learn labels through LDP or some other MPLS applications. You can use MPLS LSP ping or traceroute to validate an LSP used for forwarding traffic for a given FEC.

You can ensure that the router forwards MPLS packets for IPv4 FEC prefixes advertised by LDP.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask-length* [**repeat** *count*] [**exp** *exp-bits*] [**verbose**]
   or
   traceroute mpls ipv4 *destination-address*/*destination-mask-length*

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask-length* [**repeat** *count*] [**exp** *exp-bits*] [**verbose**] or **traceroute mpls ipv4** *destination-address*/*destination-mask-length* [***exp*** *exp-bits*] [***verbose***] **Example:** `switch# ping mpls ipv4 10.131.191.252/32 repeat 5 exp 5 verbose` or **Example:** `switch# traceroute mpls ipv4 10.131.191.252/32`	Selects an LDP IPv4 prefix FEC for validation. **Note** Cisco NX-OS does support the return of EXP settings from the transit routers. Those values are always reported as 0. If you enter a command with an **exp** option, for example, exp 5, the output will always display 0 for the EXP bit settings reported from the responding routers.

Validating a Layer 2 FEC

SUMMARY STEPS

1. **ping mpls pseudowire** *ipv4-address vc-id*

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls pseudowire** *ipv4-address vc-id* **Example:** `Switch # ping mpls pseudowire 10.131.191.252 333`	Selects a Layer 2 FEC for validation.

Using DSCP to Request a Specific Class of Service in an Echo Reply

The reply DSCP option is supported in the experimental mode for IETF draft-ietf-mpls-lsp-ping-03.txt. Cisco implemented a vendor-specific extension for the reply DSCP option rather than using a Reply ToS TLV. A Reply ToS TLV serves the same purpose as the **reply dscp** command in RFC 4379. This draft provides a standardized method of controlling the reply DSCP.

Note Before RFC 4379, Cisco implemented the Reply DSCP option as an experimental capability using a Cisco vendor extension TLV. If a router is configured to encode MPLS echo packets for draft Version 3 implementations, a Cisco vendor extension TLV is used instead of the Reply ToS TLV that was defined in RFC 4379.

When Cisco NX-OS is configured to operate in revision 3 mode, it will still send using the Cisco vendor extension TLV. The software will also respond to any echo request packets with these TLVs.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask-length* [**reply dscp** *dscp-value*]
   or
   traceroute mpls ipv4 *destination-address*/*destination-mask-length* [**reply dscp** *dscp-value*]

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask-length* [**reply dscp** *dscp-value*] or **traceroute mpls ipv4** *destination-address*/*destination-mask-length* [**reply dscp** *dscp-value*] **Example:** `switch# ping mpls ipv4 10.131.191.252/32 reply dscp 50` or **Example:** `switch# traceroute mpls ipv4 10.131.191.252/32 reply dscp 50`	Controls the DSCP value of an echo reply.

Controlling How a Responding Router Replies to an MPLS Echo Request

The reply mode controls how a responding router replies to an MPLS echo request when you enter the **ping mpls** or **traceroute mpls** command. There are two reply modes for an echo request packet:

- ipv4—Reply with an IPv4 UDP packet (default)
- router-alert—Reply with an IPv4 UDP packet with router alert

Note You should use ipv4 and router-alert reply modes with each other to prevent false negatives. If you do not receive a reply through the ipv4 mode, send a test with the router-alert reply mode. If both fail, that means that something is wrong in the return path. The problem may be only that the Reply ToS is not set correctly.

This section includes the following topics:

ipv4 Reply Mode

An IPv4 packet is the most common reply mode used with the **ping mpls** or **traceroute mpls** command when you want to periodically poll the integrity of an LSP. With this option, you do not have explicit control over whether the packet traverses IP or MPLS hops to reach the originator of the MPLS echo request. If the originating (head-end) router fails to receive a reply to an MPLS echo request when you use the **reply mode ipv4** keywords, use the **reply mode router-alert** keywords.

Router-Alert Reply Mode

The router-alert reply mode adds the router alert option to the IP header. When an IP packet that contains an IP router alert option in its IP header or an MPLS packet with a router alert label as its outermost label arrives at a router, the router punts (redirects) the packet to the supervisor level for handling to force the Cisco router to handle the packet at each intermediate hop as it moves back to the destination. Hardware and line-card forwarding inconsistencies are bypassed. Router-alert reply mode is more expensive than IPv4 mode because the reply goes hop by hop. It is also slower, so the sender receives a reply in a relatively longer period of time.

Table 34-4 describes how IP and MPLS packets with an IP router alert option are handled by the router switching path processes.

Table 34-4 Path Process Handling of IP and MPLS Router Alert Packets

Incoming Packet	Software Switching Action	Outgoing Packet
IP packet—Router alert option in IP header	Forwards the packet as is	IP packet—Router alert option in IP header
	Forwards the packet as is	MPLS packet
MPLS packet—Outermost label contains a router alert	Removes the outermost router alert label and forwards the packet as an IP packet	IP packet—Router alert option in IP header

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address/destination-mask-length* **reply mode** {**ipv4** | **router-alert**}
   or
   traceroute mpls ipv4 *destination-address/destination-mask* **reply mode** {**ipv4** | **router-alert**}

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask-length* **reply mode** {**ipv4** | **router-alert**} or **traceroute mpls ipv4** *destination-address*/*destination-mask* **reply mode** {**ipv4** | **router-alert**} **Example:** `switch# ping mpls ipv4 10.131.191.252/32 reply mode ipv4` or **Example:** `switch# traceroute mpls ipv4 10.131.191.252/32 reply mode router-alert`	Checks MPLS LSP connectivity. or Discovers MPLS LSP routes that packets actually take when traveling to their destinations. **Note** To specify the reply mode, you must enter the **reply mode** keyword with the **ipv4** or **router-alert** keyword.

Preventing Loops When Using MPLS LSP Ping and LSP Traceroute Command Options

The interaction of the MPLS Embedded Management—LSP Ping for LDP feature options can cause loops. See the following topics for a description of the loops you may encounter with the **ping mpls** and **traceroute mpls** commands:

- Using MPLS LSP Ping to Discover Possible Loops, page 34-77
- Using MPLS LSP Traceroute to Discover Possible Loops, page 34-78

Using MPLS LSP Ping to Discover Possible Loops

With the MPLS LSP Ping feature, loops can occur if you use the UDP destination address range, repeat option, or sweep option.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* [**destination** *address-start address-end increment*] [**repeat** *count*] [**sweep** *minimum maximum size-increment*]

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* [**destination** *address-start address-end increment*] [**repeat** *count*] [**sweep** *minimum maximum size-increment*] **Example:** `switch# ping mpls ipv4 10.131.159.251/32 destination 127.0.0.1 127.0.0.2 1 repeat 2 sweep 1450 1475 25`	Checks MPLS LSP connectivity.

Using MPLS LSP Traceroute to Discover Possible Loops

With the MPLS LSP Traceroute feature, loops can occur if you use the UDP destination address range option and the Time-to-Live (TTL) option.

By default, the maximum TTL is set to 30. Therefore, the traceroute output might contain 30 lines if the target of the traceroute is not reached, which can happen when an LSP problem exists. If an LSP problem occurs, there might be duplicate entries. The router address of the last point that the trace reaches is repeated until the output is 30 lines. You can ignore the duplicate entries.

SUMMARY STEPS

1. **traceroute mpls ipv4** *destination-address*/*destination-mask* [**destination** *address-start address-end address-increment*] [**ttl** *maximum-time-to-live*]

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls ipv4** *destination-address*/*destination-mask* [**destination** *address-start address-end address increment*] [**ttl** *maximum-time-to-live*] **Example:** `switch# traceroute mpls ipv4 10.131.159.251/32 destination 127.0.0.1 127.0.0.3 1 ttl 5`	Discovers MPLS LSP routes that packets take when traveling to their destinations. The example shows how a loop can occur.

Detecting LSP Breaks

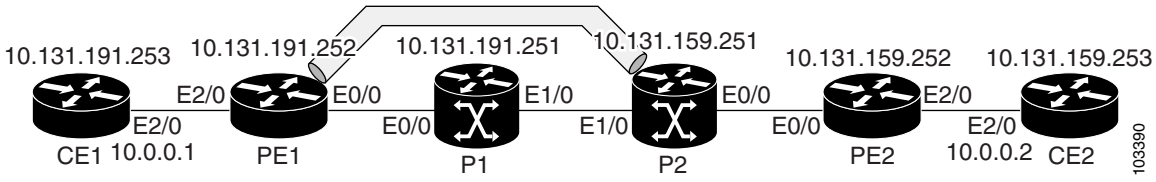
If there is a problem forwarding MPLS packets in your network, you can determine where the LSP breaks are. This section describes the maximum transmission unit (MTU) discovery in an LSP.

Untagged output interfaces at a penultimate hop do not impact the forwarding of IP packets through an LSP because the forwarding decision is made at the penultimate hop by using an incoming label. However, untagged output interfaces cause MPLS VPN traffic to be dropped at the penultimate hop.

During an MPLS LSP ping, MPLS echo request packets are sent with the IP packet attribute set to the Don't Fragment (DF) bit in the IP header of the packet. This process allows you to use the MPLS echo request to test for the MTU that can be supported for the packet through LSP without fragmentation.

The following figure shows a sample network with a single LSP from PE1 to PE2 formed with labels advertised by the LDP.

Figure 34-4 Sample Network with LSP—Labels Advertised by LDP



You can determine the maximum receive unit (MRU) at each hop by using the MPLS LSP traceroute to trace the LSP. The MRU is the maximum size of a labeled packet that can be forwarded through an LSP.

This section includes the following topics:

Tracking Packets Tagged as Implicit Null

You can track packets tagged as implicit null.

SUMMARY STEPS

1. **traceroute mpls ipv4** *destination-address/destination-mask*

DETAILED STEPS

	Command	Purpose
Step 1	**traceroute mpls ipv4** *destination-address/destination-mask* **Example:** `switch# traceroute mpls ipv4 10.131.159.252/32`	Discovers MPLS LSP routes that packets actually take when traveling to their destinations.

Determining Why a Packet Could Not Be Sent

The Q return code means that the packet could not be sent. The problem can be caused by insufficient processing memory, but it probably results because an LSP could not be found that matches the FEC information that was entered on the command line.

You must determine the reason why the packet was not forwarded so that you can fix the problem in the path of the LSP. To do so, look at the Routing Information Base (RIB), the Forwarding Information Base (FIB), the Label Information Base (LIB), and the MPLS LFIB. If there is no entry for the FEC in any of these routing or forwarding bases, you see a Q return code.

SUMMARY STEPS

1. **show ip route** [*network*[*/length*] **detail**
2. **show mpls switching** [*network*[*/length*]]
3. **attach module** *module_number*
4. **show forwarding** [**route | mpls | adjacency mpls stats**]

DETAILED STEPS

	Command	Purpose
Step 1	**show ip route** [*network*[*/length*]] **detail** **Example:** `switch# show ip route 10.137.191.252 detail`	Displays the current state of the routing table. When the MPLS echo reply returns a Q, troubleshooting occurs on the routing information database.
Step 2	**show mpls switching** [*network*[*/length*]] **Example:** `switch# show mpls switching`	Displays the contents of the MPLS LFIB. Packets that are label switched use the MPLSFWD component forwarding tables.
Step 3	**attach module** *module_number* **Example:** `switch# attach module 1`	Attaches to the linecard module.
Step 4	**show forwarding** [**route** | **mpls** | **adjacency mpls stats**] **Example:** `module-1# show forwarding route` `module-1# show forwarding mpls` `module-1# show forwarding adjacency mpls stats`	Displays the contents of the IPFIB. This table shows the packets that are IP or label switched by the linecard hardware.

Detecting LSP Breaks when Load Balancing Is Enabled for IPv4 LDP LSPs

An ICMP ping or trace follows one path from the originating router to the target router. Round robin load balancing of IP packets from a source router discovers the various output paths to the target IP address.

For MPLS LSP ping and traceroute, Cisco routers use the source and destination addresses in the IP header for load balancing when multiple paths exist through the network to a target router. The Cisco implementation of MPLS might check the destination address of an IP payload to accomplish load balancing.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask-length* [**destination** *address-start address-end increment*]

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask-length* [**destination** *address-start address-end increment*] **Example:** `switch# ping mpls ipv4 10.131.159.251/32 destination 127.0.0.1/8`	Checks for load balancing paths. The value of the destination address is 127.*z.y.x*/8.

Specifying the Interface Through Which Echo Packets Leave a Router

You can control the interface through which packets leave a router. Path output information is used as input to LSP ping and traceroute.

The echo request output interface control feature allows you to force echo packets through the paths that perform detailed debugging or characterizing of the LSP. This feature is useful if a PE router connects to an MPLS cloud and there are broken links. You can direct traffic through a certain link. The feature also is helpful for troubleshooting network problems.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* [**output interface** *tx-interface*]
   or
   traceroute mpls ipv4 *destination-address*/*destination-mask* [**output interface** *tx-interface*]

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* [**output interface** *tx-interface*] or **traceroute mpls ipv4** *destination-address*/*destination-mask* [**output interface** *tx-interface*] **Example:** `switch# ping mpls ipv4 10.131.159.251/32 output interface ethernet0/0` or **Example:** `switch# traceroute mpls ipv4 10.131.159.251/32 output interface ethernet0/0`	Checks MPLS LSP connectivity. or Discovers MPLS LSP routes that packets take when traveling to their destinations. **Note** For this task, you must specify the **output interface** keyword.

Pacing the Transmission of Packets

Echo request traffic pacing allows you to pace the transmission of packets so that the receiving router does not drop packets.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* [**interval** *ms*]
   or
   traceroute mpls ipv4 *destination-address*/*destination-mask*

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* [**interval** *ms*] or **traceroute mpls ipv4** *destination-address*/*destination-mask* **Example:** `switch# ping mpls ipv4 10.131.159.251/32 interval 2` or **Example:** `switch# traceroute mpls ipv4 10.131.159.251/32`	Checks MPLS LSP connectivity. or Discovers MPLS LSP routes that packets take when traveling to their destinations. **Note** In this task, if you use the **ping mpls** command, you must specify the **interval** keyword.

Interrogating the Transit Router for Its Downstream Information by Using Echo Request request-dsmap

When you use the echo request request-dsmap capability troubleshooting feature with the TTL flag, you can selectively interrogate a transit router. If there is a failure, you do not have to enter an **lsp traceroute** command for each previous failure; you can focus just on the failed hop.

A request-dsmap flag in the downstream mapping flags field and procedures that specify how to trace noncompliant routers allow you to arbitrarily TTL expire MPLS echo request packets with a wildcard downstream map (DSMAP).

Echo request DSMAPs received without labels indicate that the sender did not have any DSMAPs to validate. If the downstream router ID field of the DSMAP TLV in an echo request is set to the ALLROUTERs address (224.0.0.2) and there are no labels, the source router can arbitrarily query a transit router for its DSMAP information.

Use the **ping mpls** command to allow an MPLS echo request to be TTL-expired at a transit router with a wildcard DSMAP for the explicit purpose of troubleshooting and querying the downstream router for its DSMAPs. The default is that the DSMAP has an IPv4 bitmap hashkey. You also can select hashkey 0 (none). The **ping mpls** command allows the source router to selectively TTL expire an echo request at a transit router to interrogate the transit router for its downstream information. The ability to select a multipath (hashkey) type allows the transmitting router to interrogate a transit router for load-balancing information as is done with multipath LSP traceroute but without having to interrogate all subsequent nodes traversed between the source router and the router on which each echo request TTL expires. You should use an echo request with the TTL setting because if an echo request arrives at the egress of the LSP without an echo request, the responding routers never return DSMAPs.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* [**dsmap** [**hashkey** {**none** | **ipv4 bitmap** *bitmap-size*}]]

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* [**dsmap** [**hashkey** {**none** | **ipv4 bitmap** *bitmap-size*}]] **Example:** `switch# ping mpls ipv4 10.161.251/32 dsmap hashkey ipv4 bitmap 16`	Checks MPLS LSP connectivity. **Note** In this task, you must specify the **dsmap** and **hashkey** keywords.

Interrogating a Router for its DSMAP

The router can interrogate the software or hardware forwarding layer for the depth limit that needs to be returned in the DSMAP TLV. If forwarding does not provide a value, the default is 255.

To determine the depth limit, specify the **dsmap** and **ttl** keywords in the **ping mpls** command. The transit router is interrogated for its DSMAP. The depth limit is returned with the echo reply DSMAP. A value of 0 means that the IP header is used for load balancing. Another value indicates that the IP header load balances up to the specified number of labels.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* **ttl** *time-to-live* **dsmap**

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* **ttl** *time-to-live* **dsmap** **Example:** `switch# ping mpls ipv4 10.131.159.252/32 ttl 1 dsmap`	Checks MPLS LSP connectivity. **Note** You must specify the **ttl** and **dsmap** keywords.

Requesting That a Transit Router Validate the Target FEC Stack

An MPLS echo request tests a particular LSP. The LSP to be tested is identified by the FEC stack.

To request that a transit router validate the target FEC stack, set the V flag from the source router by entering the **flags fec** keyword in the **ping mpls** and **traceroute mpls** commands. The default is that echo request packets are sent with the V flag set to 0.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* **flags fec**
   or
   traceroute mpls ipv4 *destination-address*/*destination-mask* **flags fec**

DETAILED STEPS

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* **flags fec** or **traceroute mpls ipv4** *destination-address*/*destination-mask* **flags fec** **Example:** `switch# ping mpls ipv4 10.131.159.252/32 flags fec` or **Example:** `switch# traceroute mpls ipv4 10.131.159.252/32 flags fec`	Checks MPLS LSP connectivity. or Discovers MPLS LSP routes that packets actually take when traveling to their destinations. **Note** You must enter the **flags fec** keyword.

Using LSP Ping to Detect LSP Breakages Caused by Untagged Interfaces

For MPLS LSP ping and traceroute of LSPs carrying IPv4 FECs, you can force an explicit null label to be added to the MPLS label stack even though the label was unsolicited. This process allows LSP ping to detect LSP breakages caused by untagged interfaces. LSP ping does not report that an LSP is operational when it is unable to send MPLS traffic.

An explicit null label is added to an MPLS label stack if MPLS echo request packets are forwarded from untagged interfaces that are directly connected to the destination of the LSP ping or if the IP TTL value for the MPLS echo request packets is set to 1.

When you enter the **lsp ping** command, you are testing the LSP's ability to carry IP traffic. Failures at untagged output interfaces at the penultimate hop are not detected. Explicit-null shimming allows you to test an LSP's ability to carry MPLS traffic.

You can enable LSP ping to detect LSP breakages caused by untagged interfaces by specifying the **force-explicit-null** keyword in the **ping mpls** or **traceroute mpls** commands.

SUMMARY STEPS

1. **ping mpls ipv4** *destination-address*/*destination-mask* **force-explicit-null**
   or
   traceroute mpls ipv4 *destination-address*/*destination-mask* **force-explicit-null**

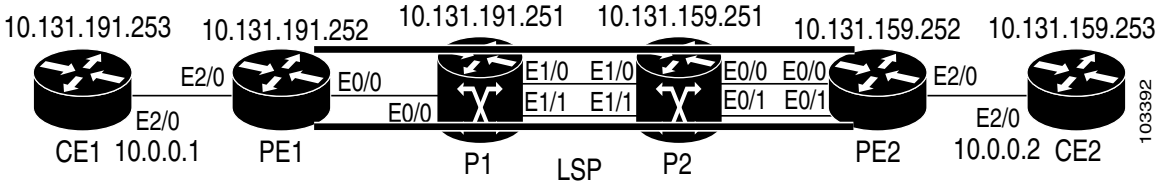
DETAILED STEP

	Command	Purpose
Step 1	**ping mpls ipv4** *destination-address*/*destination-mask* **force-explicit-null** or **traceroute mpls ipv4** *destination-address*/*destination-mask* **force-explicit-null** **Example:** `switch# ping mpls ipv4 10.131.191.252/32 force-explicit-null` or **Example:** `switch# traceroute mpls ipv4 10.131.191.252/32 force-explicit-null`	Checks MPLS LSP connectivity. or Discovers MPLS LSP routes that packets actually take when traveling to their destinations. **Note** You must enter the **force-explicit-null** keyword.

Troubleshooting Examples Using MPLS LSP Ping and Traceroute

Examples for the MPLS LSP ping and traceroute for LDP and TE and LSP ping for virtual circuit connection verification (VCCV) are based on the sample topology shown in the figure below.

Figure 34-5 Sample Topology for Troubleshooting Examples



This section includes the following topics:

- Example: Enabling Compatibility Between the MPLS LSP and Ping or Traceroute Implementation, page 34-86
- Example: Validating an FEC by Using MPLS LSP Ping and LSP Traceroute, page 34-86
- Example: Validating a Layer 2 FEC by Using MPLS LSP Ping, page 34-87
- Example: Using DSCP to Request a Specific Class of Service in an Echo Reply, page 34-87
- Example: Controlling How a Responding Router Replies to an MPLS Echo Request, page 34-87
- Example: Preventing Loops when Using MPLS LSP Ping and LSP Traceroute Command Options, page 34-88
- Example: Detecting LSP Breaks, page 34-91

Example: Enabling Compatibility Between the MPLS LSP and Ping or Traceroute Implementation

The following example shows how to configure MPLS multipath LSP traceroute to interoperate with a vendor implementation that does not interpret RFC 4379 as Cisco NX-OS does:

```
configure terminal
!
mpls oam
 echo revision 4
 no echo vendor-extension
 exit
```

The default echo revision number is 4, which corresponds to the IEFT draft 11.

Example: Validating an FEC by Using MPLS LSP Ping and LSP Traceroute

The following example shows how to use the **ping mpls** command to test connectivity of an IPv4 LDP LSP:

```
switch# ping mpls ipv4 10.137.191.252/32 repeat 1

Sending 1, 100-byte MPLS Echos to 10.137.191.252/32,
timeout is 2 seconds, send interval is 0 msec:
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'L' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'X' - unknown return code, 'x' - return code 0
```

Example: Validating a Layer 2 FEC by Using MPLS LSP Ping

The following example shows how to validate a Layer 2 FEC:

```
Switch# ping mpls pseudowire 10.10.10.15 108 vc-id 333
Sending 5, 100-byte MPLS Echos to 10.10.10.15,
timeout is 2 seconds, send interval is 0 msec:
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'I' - unknown upstream index,
    'X' - unknown return code, 'x' - return code 0
Type escape sequence to abort.
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/32/40 ms PE-802#
```

Example: Using DSCP to Request a Specific Class of Service in an Echo Reply

The following example shows how to use DSCP to request a specific CoS in an echo reply:

```
switch# ping mpls ipv4 10.131.159.252/32 reply dscp 50

 <0-63>   Differentiated services codepoint value
 af11     Match packets with AF11 dscp (001010)
 af12     Match packets with AF12 dscp (001100)
 af13     Match packets with AF13 dscp (001110)
 af21     Match packets with AF21 dscp (010010)
 af22     Match packets with AF22 dscp (010100)
 af23     Match packets with AF23 dscp (010110)
 af31     Match packets with AF31 dscp (011010)
 af32     Match packets with AF32 dscp (011100)
 af33     Match packets with AF33 dscp (011110)
 af41     Match packets with AF41 dscp (100010)
 af42     Match packets with AF42 dscp (100100)
 af43     Match packets with AF43 dscp (100110)
 cs1      Match packets with CS1(precedence 1) dscp (001000)
 cs2      Match packets with CS2(precedence 2) dscp (010000)
 cs3      Match packets with CS3(precedence 3) dscp (011000)
 cs4      Match packets with CS4(precedence 4) dscp (100000)
 cs5      Match packets with CS5(precedence 5) dscp (101000)
 cs6      Match packets with CS6(precedence 6) dscp (110000)
 cs7      Match packets with CS7(precedence 7) dscp (111000)
 default  Match packets with default dscp (000000)
 ef       Match packets with EF dscp (101110)
```

Example: Controlling How a Responding Router Replies to an MPLS Echo Request

The following example shows how to check MPLS LSP connectivity by using ipv4 reply mode:

```
switch# ping mpls ipv4 10.131.191.252/32 reply mode ipv4
```

Example: Preventing Loops when Using MPLS LSP Ping and LSP Traceroute Command Options

This section contains the following topics:

Example: Possible Loops with MPLS LSP Ping

The following example shows how a loop operates if you use the following **ping mpls** command:

```
switch# ping mpls ipv4 10.131.159.251/32 destination 127.0.0.1 127.0.0.2 1 repeat 2
sweep 1450 1475 25

Sending 2, [1450..1500]-byte MPLS Echos to 10.131.159.251/32,
     timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
Destination address 127.0.0.1
!
!
Destination address 127.0.0.2
!
!
Destination address 127.0.0.1
!
!
Destination address 127.0.0.2
!
!
```

Entering the **ping mpls** command enables the router to send each packet size range for each destination address until the end address is reached. For this example, the loop continues in the same manner until the destination address, 127.0.0.5, is reached. The sequence continues until the number is reached that you specified with the **repeat** *count* keyword and argument. For this example, the repeat count is 2. The MPLS LSP ping loop sequence is as follows:

```
repeat  = 1
  destination address 1 (address-start)
    for (size from sweep minimum to maximum, counting by size-increment)
      send an lsp ping

  destination address 2 (address-start + address-increment)
    for (size from sweep minimum to maximum, counting by size-increment)
      send an lsp ping

  destination address 3 (address-start + address-increment + address-increment)
     for (size from sweep minimum to maximum, counting by size-increment)
      send an lsp ping
```

```
  .
  .
  .
  until destination address = address-end

  .
  .
  .
  until repeat = count 2
```

Example: Possible Loop with MPLS LSP Traceroute

The following example shows how a loop occurs if you use the following **traceroute mpls** command:

```
switch# traceroute mpls ipv4 10.131.159.251/32 destination 127.0.0.1 127.0.0.3 1 ttl 5

Tracing MPLS Label Switched Path to 10.131.159.251/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
Destination address 127.0.0.1
  0 10.131.191.230 MRU 1500 [Labels: 19 Exp: 0]
R 1 10.131.159.226 MRU 1504 [implicit-null] 40 ms
! 2 10.131.159.225 40 ms
Destination address 127.0.0.2
  0 10.131.191.230 MRU 1500 [Labels: 19 Exp: 0]
R 1 10.131.159.226 MRU 1504 [implicit-null] 40 ms
! 2 10.131.159.225 40 ms
Destination address 127.0.0.3
  0 10.131.191.230 MRU 1500 [Labels: 19 Exp: 0]
R 1 10.131.159.226 MRU 1504 [implicit-null] 40 ms
! 2 10.131.159.225 48 ms
```

Entering the **mpls trace** command enables the router to send each TTL from 1 to the maximum TTL (**ttl** *maximum-time-to-live* keyword and argument) for each destination address until the address specified with the destination *end-address* argument is reached. In this example, the maximum TTL is 5 and the end destination address is 127.0.0.3. The MPLS LSP traceroute loop sequence is as follows:

```
destination address 1 (address-start)
  for (ttl from 1 to maximum-time-to-live)
    send an lsp trace

destination address 2 (address-start + address-increment)
  for (ttl from 1 to 5)
    send an lsp trace

destination address 3 (address-start + address-increment + address-increment)
  for (ttl from 1 to maximum-time-to-live)
    send an lsp trace
.
.
.
until destination address = 4
```

The following example shows that the trace encountered an LSP problem at the router that has an IP address of 10.6.1.6:

```
switch# traceroute mpls ipv4 10.6.7.4/32

Tracing MPLS Label Switched Path to 10.6.7.4/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.6.1.14 MRU 4470 [Labels: 22 Exp: 0]
R 1 10.6.1.5 MRU 4470 [Labels: 21 Exp: 0] 2 ms
R 2 10.6.1.6 4 ms                  <------ Router address repeated for 2nd to 30th TTL.
R 3 10.6.1.6 1 ms
R 4 10.6.1.6 1 ms
R 5 10.6.1.6 3 ms
R 6 10.6.1.6 4 ms
R 7 10.6.1.6 1 ms
R 8 10.6.1.6 2 ms
R 9 10.6.1.6 3 ms
R 10 10.6.1.6 4 ms
R 11 10.6.1.6 1 ms
R 12 10.6.1.6 2 ms
R 13 10.6.1.6 4 ms
R 14 10.6.1.6 5 ms
R 15 10.6.1.6 2 ms
R 16 10.6.1.6 3 ms
R 17 10.6.1.6 4 ms
R 18 10.6.1.6 2 ms
R 19 10.6.1.6 3 ms
R 20 10.6.1.6 4 ms
R 21 10.6.1.6 1 ms
R 22 10.6.1.6 2 ms
R 23 10.6.1.6 3 ms
R 24 10.6.1.6 4 ms
R 25 10.6.1.6 1 ms
R 26 10.6.1.6 3 ms
R 27 10.6.1.6 4 ms
R 28 10.6.1.6 1 ms
R 29 10.6.1.6 2 ms
R 30 10.6.1.6 3 ms                    <------ TTL 30.
```

If you know the maximum number of hops in your network, you can set the TTL to a lower value with the **traceroute mpls ttl** *maximum-time-to-live* command. The following example shows the same **traceroute** command as the previous example, except that this time, the TTL is set to 5:

```
switch# traceroute mpls ipv4 10.6.7.4/32 ttl 5

Tracing MPLS Label Switched Path to 10.6.7.4/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0
```

```
Type escape sequence to abort.
  0 10.6.1.14 MRU 4470 [Labels: 22 Exp: 0]
R 1 10.6.1.5 MRU 4474 [No Label] 3 ms
R 2 10.6.1.6 4 ms                  <------ Router address repeated for 2nd to 5th TTL.
R 3 10.6.1.6 1 ms
R 4 10.6.1.6 3 ms
R 5 10.6.1.6 4 ms
```

Example: Detecting LSP Breaks

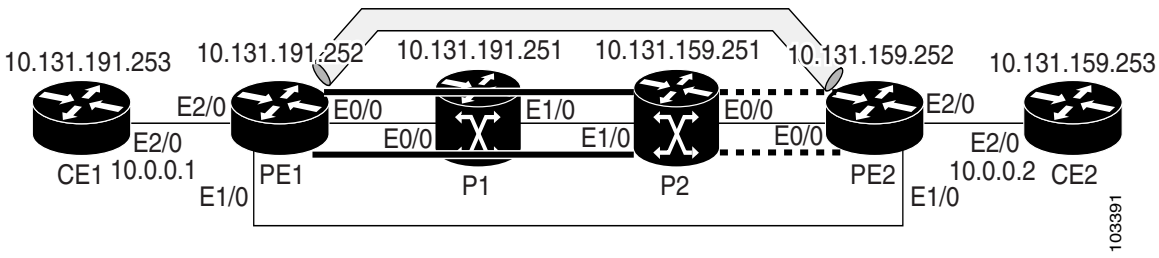
This section includes the following topics:

Example: Troubleshooting with LSP Ping or Traceroute

ICMP **ping** and **trace** commands are often used to help diagnose the root cause of a failure. When an LSP is broken, the packet might reach the target router by IP forwarding, which makes the ICMP ping and traceroute features unreliable for detecting MPLS forwarding problems. The MPLS LSP ping or traceroute features extend this diagnostic and troubleshooting ability to the MPLS network and handle inconsistencies (if any) between the IP and MPLS forwarding tables, inconsistencies in the MPLS control and data plane, and problems with the reply path.

The following figure shows a sample topology with an LDP LSP.

Figure 34-6 Sample Topology with LDP LSP



This section includes the following topics:

Verifying that the LSP Is Configured Correctly

Use the output from the **show** commands in this section to verify that the LSP is configured correctly.

The following example shows that tunnel 1 is in the MPLS forwarding table:

```
PE1# show mpls forwarding-table 10.131.159.252

Local  Outgoing    Prefix            Bytes tag  Outgoing   Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
22     18 [T] 10.131.159.252/32 0          Tu1        point2point


[T]     Forwarding through a TSP tunnel.
        View additional tagging info with the 'detail' option
```

The following example shows that the **traceroute mpls** command issued at PE1 verifies that packets with 16 as the outermost label and 18 as the end-of-stack label are forwarded from PE1 to PE2:

```
PE1# traceroute mpls ipv4 10.131.159.252/32

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
   0 10.131.191.252 MRU 1496 [Labels: 16/18 Exp: 0/0] L 1 10.131.191.229
   MRU 1508 [Labels: 18 Exp: 0] 0 ms L 2 10.131.159.225
   MRU 1504 [Labels: implicit-null Exp: 0] 0 ms ! 3 10.131.159.234 20 ms
   PE1#
```

The MPLS LSP traceroute to PE2 is successful as indicated by the exclamation point (!).

Discovery of LSP Breaks

Use the output of the commands in this section to discover LSP breaks.

The following example shows that an LDP target session is established between routers PE1 and P2:

```
PE1# show mpls ldp discovery

 Local LDP Identifier:
    10.131.191.252:0
    Discovery Sources:
    Interfaces:
        Ethernet0/0 (ldp): xmit/recv
            LDP Id: 10.131.191.251:0
        Tunnel1 (ldp): Targeted -> 10.131.159.251
    Targeted Hellos:
        10.131.191.252 -> 10.131.159.252 (ldp): active/passive, xmit/recv
            LDP Id: 10.131.159.252:0
        10.131.191.252 -> 10.131.159.251 (ldp): active, xmit/recv
            LDP Id: 10.131.159.251:0
```

The following example shows the P2 router in global configuration mode:

```
P2(config)# no mpls ldp discovery targeted-hello accept
```

The LDP configuration change causes the targeted LDP session between the head-end and tail-end of the TE tunnel to go down. Labels for IPv4 prefixes learned by P2 are not advertised to PE1. All IP prefixes reachable by P2 are reachable by PE1 only through IP (not MPLS). Packets destined for those prefixes through Tunnel 1 at PE1 will be IP switched at P2 (which is undesirable).

The following example shows that the LDP targeted session is down:

```
PE1# show mpls ldp discovery

 Local LDP Identifier:
    10.131.191.252:0
    Discovery Sources:
    Interfaces:
        Ethernet0/0 (ldp): xmit/recv
            LDP Id: 10.131.191.251:0
        Tunnel1 (ldp): Targeted -> 10.131.159.251
    Targeted Hellos:
        10.131.191.252 -> 10.131.159.252 (ldp): active/passive, xmit/recv
            LDP Id: 10.131.159.252:0
        10.131.191.252 -> 10.131.159.251 (ldp): active, xmit
```

Cisco NX-OS has three components that store information related to forwarding:

1. The IPv4 routing component.
2. The MPLS forwarding component for label switched packets.
3. The packets that are IP or label switched by the line card hardware.

The following commands allow you to display the tables stored by each of these components:

```
PE1# show ip route 10.137.191.252 detail
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]

10.137.191.252/32, ubest/mbest: 1/0
    *via 10.133.191.246, Eth1/1, [115/100], 2d17h, isis-p1, L2 (mpls)
      MPLS[0]: Label=20 E=0 TTL=255 S=0, LDP
      client-specific data: 42

PE1# show mpls switching

Legend:
(P)=Protected, (F)=FRR active, (*)=more labels in stack.

In-Label Out-Label FEC name Out-Interface Next-Hop
VRF default
20  3 10.131.191.252/32 Eth1/1 10.133.191.246
17  3 10.132.191.252/32 Eth1/2 10.132.191.225
21 19 10.136.191.252/32 Eth1/1 10.133.191.246
18 20 10.137.191.252/32 Eth1/1 10.133.191.246

PE1# attach module 1
module-1# show forwarding route
IPv4 routes for table default/base
------------------+------------------+----------------------+-----------------
Prefix            | Next-hop         | Interface            | Labels
------------------+------------------+----------------------+-----------------
0.0.0.0/32          Drop               Null0
```

```
127.0.0.0/8         Drop               Null0
255.255.255.255/32  Receive            sup-eth1
10.131.191.252/32   10.133.191.246     Ethernet1/1            NO-OP
10.132.191.224/30   Attached           Ethernet1/2
10.132.191.224/32   Drop               Null0
10.132.191.225/32   10.132.191.225     Ethernet1/2
10.132.191.226/32   Receive            sup-eth1
10.132.191.227/32   Attached           Ethernet1/2
10.132.191.228/30   10.133.191.246     Ethernet1/1
10.132.191.252/32   10.132.191.225     Ethernet1/2            NO-OP
10.133.191.244/30   Attached           Ethernet1/1
10.133.191.244/32   Drop               Null0
10.133.191.245/32   Receive            sup-eth1
10.133.191.246/32   10.133.191.246     Ethernet1/1
10.133.191.247/32   Attached           Ethernet1/1
10.133.191.252/32   Receive            sup-eth1
10.136.191.244/30   10.133.191.246     Ethernet1/1
10.136.191.252/32   10.133.191.246     Ethernet1/1            PUSH 19
10.137.191.112/30   10.133.191.246     Ethernet1/1
10.137.191.252/32   10.133.191.246     Ethernet1/1            PUSH 20
module-1# show forwarding mpls
--------+-----------+------------------+----------------+-------------+-------
Local   |Prefix     |FEC               |Next-Hop        |Interface    |Out
Label   |Table Id   |(Prefix/Tunnel id)|                |             |Label
--------+-----------+------------------+----------------+-------------+-------
20      |0x1        |10.131.191.252/32 |10.133.191.246  |Ethernet1/1  |3
17      |0x1        |10.132.191.252/32 |10.132.191.225  |Ethernet1/2  |3
21      |0x1        |10.136.191.252/32 |10.133.191.246  |Ethernet1/1  |19
18      |0x1        |10.137.191.252/32 |10.133.191.246  |Ethernet1/1  |20

module-1# show forwarding adjacency mpls stats
next-hop         rewrite info   tx packets    tx bytes       Label info
-----------------------------------------------------------------------
10.133.191.246   Ethernet1/1    0             0              NO-OP 3
10.133.191.246   Ethernet1/1    0             0              POP 3
10.133.191.246   Ethernet1/1    0             0              PUSH 19
10.133.191.246   Ethernet1/1    0             0              SWAP 19
10.133.191.246   Ethernet1/1    0             0              PUSH 20
10.133.191.246   Ethernet1/1    0             0              SWAP 20
10.132.191.225   Ethernet1/2    0             0              NO-OP 3
10.132.191.225   Ethernet1/2    2             140            POP 3
module-1# exit
PE1#
```

The following example shows the PE1 router:

```
PE1# ping mpls ipv4 10.131.159.252/32 repeat 1

Sending 1, 100-byte MPLS Echos to 10.131.159.252/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
R
Success rate is 0 percent (0/1)
```

The **ping mpls** command fails. The R indicates that the sender of the MPLS echo reply had a routing entry but no MPLS FEC. Entering the **verbose** keyword with the **ping mpls** command displays the MPLS LSP echo reply sender address and the return code. You should be able to determine where the breakage occurred by using Telnet to the replying router and inspecting its forwarding and label tables. You might need to look at the neighboring upstream router as well, because the breakage might be on the upstream router.

```
PE1# ping mpls ipv4 10.131.159.252/32 repeat 1 verbose

Sending 1, 100-byte MPLS Echos to 10.131.159.252/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
R   10.131.159.225, return code 6

Success rate is 0 percent (0/1)
```

Alternatively, use the LSP **traceroute** command to figure out which router caused the breakage. In the following example, for subsequent values of TTL greater than 2, the same router keeps responding (10.131.159.225), which suggests that the MPLS echo request keeps getting processed by the router regardless of the TTL. Inspection of the label stack shows that P1 pops the last label and forwards the packet to P2 as an IP packet. The packet keeps getting processed by P2 because it is being forwarded. MPLS echo request packets cannot be forwarded by the destination address in the IP header because the address is set to a 127/8 address.

```
PE1# traceroute mpls ipv4 10.131.159.252/32 ttl 5

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.191.230 MRU 1496 [Labels: 22/19 Exp: 0/0]
R 1 10.131.159.226 MRU 1500 [Labels: 19 Exp: 0] 40 ms
R 2 10.131.159.229 MRU 1504 [implicit-null] 28 ms
! 3 10.131.159.230 40 ms
pe1#
```

Example: MTU Discovery in an LSP

The following example shows the results when the LSP is formed with labels created by LDP:

```
PE1# traceroute mpls ipv4 10.131.159.252/32

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes:
```

```
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.191.230 MRU 1496 [Labels: 22/19 Exp: 0/0]
R 1 10.131.159.226 MRU 1500 [Labels: 19 Exp: 0] 40 ms
R 2 10.131.159.229 MRU 1504 [implicit-null] 28 ms
! 3 10.131.159.230 40 ms
pe1#
```

To determine the MPLS MTU, first display the LSP output interface:

```
PE1# show mpls switching 10.136.191.252

Legend:
(P)=Protected, (F)=FRR active, (*)=more labels in stack.

In-Label Out-Label FEC name Out-Interface Next-Hop
21 19 10.136.191.252/32 Eth1/1 10.133.191.246

Net imposed bytes = 0 ( 1 label popped (21), 1 label pushed (19) )
```

To determine how large an echo request will fit on the LSP, first calculate the size of the IP MTU by using the **show interface** *interface-name* command as follows:

```
PE1# show interface ethernet 1/1 | include MTU
MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec
```

The IP MTU in the **show interface** *interface-name* example is 1500 bytes. Subtract the number of bytes the correspond to the label stack from the MTU number. The output of the **show mpls forwarding** command indicates that the Tag stack consists of one label (21). Therefore, the largest MPLS echo request packet that can be sent in the LSP is 1500 – (2 x 4) = 1492.

You can validate this process by using the following **ping mpls** command:

```
PE1# ping mpls ipv4 10.131.159.252/32 sweep 1492 1500 1 repeat 1

Sending 1, [1492..1500]-byte MPLS Echos to 10.131.159.252/32,
     timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
!QQQQQQQQ
Success rate is 11 percent (1/9), round-trip min/avg/max = 40/40/40 ms
```

In this command, echo packets that have a range in size from 1492 to 1500 bytes are sent to the destination address. Only packets of 1492 bytes are sent successfully, as indicated by the exclamation point (!). Packets of byte sizes 1493 to 1500 are source quenched, as indicated by the Qs.

You can pad an MPLS echo request so that a payload of a given size can be tested. The pad TLV is useful when you use the MPLS echo request to discover the MTU that is supportable by an LSP. MTU discovery is extremely important for applications like Any Transport over MPLS (AToM) that contain non-IP payloads that cannot be fragmented.

Example: Tracking Packets Tagged as Implicit Null

In the following example, Tunnel 1 is shut down, and only an LSP formed with LDP labels is established. An implicit null is advertised between the P2 and PE2 routers. Entering an MPLS LSP traceroute command at the PE1 router results in the following output that shows that packets are forwarded from P2 to PE2 with an implicit-null label. The address 10.131.159.229 is configured for the P2 Ethernet 0/0 out interface for the PE2 router.

```
PE1# traceroute mpls ipv4 10.131.159.252/32

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.191.230 MRU 1496 [Labels: 22/19 Exp: 0/0]
R 1 10.131.159.226 MRU 1500 [Labels: 19 Exp: 0] 40 ms
R 2 10.131.159.229 MRU 1504 [implicit-null] 28 ms
! 3 10.131.159.230 40 ms
pe1#
```

Example: Tracking Untagged Packets

Untagged cases are valid configurations for Interior Gateway Protocol (IGP) LSPs that could cause problems for MPLS VPNs.

Entering the **show mpls ldp discovery** command at the P2 router show that LDP is properly configured:

```
P2# show mpls ldp discovery

 Local LDP Identifier:
    10.131.159.251:0
    Discovery Sources:
    Interfaces:
        Ethernet0/0 (ldp): xmit/recv
            LDP Id: 10.131.159.252:0
        Ethernet1/0 (ldp): xmit/recv
            LDP Id: 10.131.191.251:0
```

The **show mpls ldp discovery** command output shows that Ethernet interface 0/0, which connects PE2 to P2, is sending and receiving packets.

If you enter a **no mpls ip** command on Ethernet interface 0/0, you could prevent an LDP session between the P2 and PE2 routers from being established. Entering the **show mpls ldp discovery** command on the PE router shows that the MPLS LDP session with the PE2 router is down.

```
P2# show mpls ldp discovery

 Local LDP Identifier:
```

```
    10.131.159.251:0
    Discovery Sources:
    Interfaces:
        Ethernet0/0 (ldp): xmit
        Ethernet1/0 (ldp): xmit/recv
            LDP Id: 10.131.191.251:0
```

Untagged cases would provide an MPLS LSP traceroute reply with packets tagged with No Label, as shown in the following display. You might need to reestablish an MPLS LSP session from interface P2 to PE2 by entering the **mpls ip** command on the output interface from P2 to PE2, which is Ethernet 0/0 in this example:

```
PE1# traceroute mpls ipv4 10.131.159.252/32

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.191.230 MRU 1500 [Labels: 20 Exp: 0]
R 1 10.131.159.226 MRU 1500 [Labels: 19 Exp: 0] 80 ms
R 2 10.131.159.229 MRU 1504 [No Label] 28 ms       <----No MPLS session from P2 to PE2.
! 3 10.131.159.230 40 ms
```

Example: Determining Why a Packet Could Not Be Sent

The following example shows an MPLS echo request that is not sent. The transmission failure is shown by the returned Qs.

```
PE1# ping mpls ipv4 10.0.0.1/32

Sending 5, 100-byte MPLS Echos to 10.0.0.1/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
QQQQQ
Success rate is 0 percent (0/5)
```

The following **show ip route** command demonstrate that the IPv4 address (10.0.0.1) address is not in the label forwarding information base (LFIB) or routing information base (RIB) routing table. Therefore, the MPLS echo request is not sent.

```
PE1# show ip route 10.0.0.1

% Subnet not in table
```

Example: Detecting LSP Breaks when Load Balancing Is Enabled for IPv4 LSPs

In the following examples, different paths are followed to the same destination. The output from these examples demonstrates that load balancing occurs between the originating router and the target router.

To ensure that Ethernet interface 1/0 on the PE1 router is operational, enter the following commands on the PE1 router:

```
PE1# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.

PE1(config)# interface ethernet 1/0

PE1(config-if)# no shutdown

PE1(config-if)# end

*Dec 31 19:14:10.034: %LINK-3-UPDOWN: Interface Ethernet1/0, changed state to up
*Dec 31 19:14:11.054: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet1/0,
changed state to upend
PE1#
*Dec 31 19:14:12.574: %SYS-5-CONFIG_I: Configured from console by console
*Dec 31 19:14:19.334: %OSPF-5-ADJCHG: Process 1, Nbr 10.131.159.252 on Ethernet1/0
from LOADING to FULL, Loading Done
PE1#
```

The following shows that the selected path has a path index of 0:

```
switch# ping mpls ipv4 10.131.159.251/32 destination 127.0.0.1/32

Sending 1, 100-byte MPLS Echos to 10.131.159.251/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
!
Success rate is 100 percent (1/1), round-trip min/avg/max = 40/40/40 ms
PE1#
*Dec 29 20:42:40.638: LSPV: Echo Request sent on IPV4 LSP, load_index 2,
pathindex 0, size 100
*Dec 29 20:42:40.638: 46 00 00 64 00 00 40 00 FF 11 9D 03 0A 83 BF FC
*Dec 29 20:42:40.638: 7F 00 00 01 94 04 00 00 0D AF 0D AF 00 4C 14 70
*Dec 29 20:42:40.638: 00 01 00 00 01 02 00 00 1A 00 00 1C 00 00 00 01
*Dec 29 20:42:40.638: C3 9B 10 40 A3 6C 08 D4 00 00 00 00 00 00 00 00
*Dec 29 20:42:40.638: 00 01 00 09 00 01 00 05 0A 83 9F FB 20 00 03 00
*Dec 29 20:42:40.638: 13 01 AB CD AB CD AB CD AB CD AB CD AB CD AB CD
*Dec 29 20:42:40.638: AB CD AB CD
*Dec 29 20:42:40.678: LSPV: Echo packet received: src 10.131.159.225,
dst 10.131.191.252, size 74
*Dec 29 20:42:40.678: AA BB CC 00 98 01 AA BB CC 00 FC 01 08 00 45 C0
*Dec 29 20:42:40.678: 00 3C 32 D6 00 00 FD 11 15 37 0A 83 9F E1 0A 83
*Dec 29 20:42:40.678: BF FC 0D AF 0D AF 00 28 D1 85 00 01 00 00 02 02
*Dec 29 20:42:40.678: 03 00 1A 00 00 1C 00 00 00 01 C3 9B 10 40 A3 6C
*Dec 29 20:42:40.678: 08 D4 C3 9B 10 40 66 F5 C3 C8
```

The following example shows that the selected path has a path index of 1:

```
PE1# ping mpls ipv4 10.131.159.251/32 destination 127.0.0.3/32

Sending 1, 100-byte MPLS Echos to 10.131.159.251/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes:
    '!' - success, 'Q' - request not sent, '.' - timeout,
    'L' - labeled output interface, 'B' - unlabeled output interface,
    'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
    'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
    'P' - no rx intf label prot, 'p' - premature termination of LSP,
    'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
!
Success rate is 100 percent (1/1), round-trip min/avg/max = 40/40/40 ms
PE1#
*Dec 29 20:43:09.518: LSPV: Echo Request sent on IPV4 LSP, load_index 13,
pathindex 1, size 100
*Dec 29 20:43:09.518: 46 00 00 64 00 00 40 00 FF 11 9D 01 0A 83 BF FC
*Dec 29 20:43:09.518: 7F 00 00 03 94 04 00 00 0D AF 0D AF 00 4C 88 58
*Dec 29 20:43:09.518: 00 01 00 00 01 02 00 00 38 00 00 1D 00 00 00 01
*Dec 29 20:43:09.518: C3 9B 10 5D 84 B3 95 84 00 00 00 00 00 00 00 00
*Dec 29 20:43:09.518: 00 01 00 09 00 01 00 05 0A 83 9F FB 20 00 03 00
*Dec 29 20:43:09.518: 13 01 AB CD AB CD AB CD AB CD AB CD AB CD AB CD
*Dec 29 20:43:09.518: AB CD AB CD
*Dec 29 20:43:09.558: LSPV: Echo packet received: src 10.131.159.229,
dst 10.131.191.252, size 74
*Dec 29 20:43:09.558: AA BB CC 00 98 01 AA BB CC 00 FC 01 08 00 45 C0
*Dec 29 20:43:09.558: 00 3C 32 E9 00 00 FD 11 15 20 0A 83 9F E5 0A 83
*Dec 29 20:43:09.558: BF FC 0D AF 0D AF 00 28 D7 57 00 01 00 00 02 02
*Dec 29 20:43:09.558: 03 00 38 00 00 1D 00 00 00 01 C3 9B 10 5D 84 B3
*Dec 29 20:43:09.558: 95 84 C3 9B 10 5D 48 3D 50 78
```

To see the actual path chosen, enter the **debug mpls lspv** command with the **packet** and **data** keywords.

Note The load-balancing algorithm tries to uniformly distribute packets across the available output paths by hashing based on the IP header source and destination addresses. The selection of the *address-start*, *address-end*, and *address-increment* arguments for the **destination** keyword might not provide the expected results.

Example: Specifying the Interface Through Which Echo Packets Leave a Router

The following example shows how to test load balancing from the upstream router:

```
switch# ping mpls ipv4 10.131.161.251/32 ttl 1 repeat 1 dsmap hashkey ipv4 bitmap 8

Sending 1, 100-byte MPLS Echos to 10.131.161.251/32,
     timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
```

```
L
Echo Reply received from 10.131.131.2
  DSMAP 0, DS Router Addr 10.131.141.130, DS Intf Addr 10.131.141.130
    Depth Limit 0, MRU 1500 [Labels: 54 Exp: 0]
    Multipath Addresses:
      127.0.0.3       127.0.0.5       127.0.0.7       127.0.0.8

  DSMAP 1, DS Router Addr 10.131.141.2, DS Intf Addr 10.131.141.2
    Depth Limit 0, MRU 1500 [Labels: 40 Exp: 0]
    Multipath Addresses:
      127.0.0.1       127.0.0.2       127.0.0.4       127.0.0.6
```

The following example shows how to validate that the transit router reported the proper results by determining the Echo Reply sender address two hops away and checking the rx label advertised upstream:

```
Success rate is 0 percent (0/1)

switch# traceroute mpls ipv4 10.131.161.251/32 destination 127.0.0.6 ttl 2

Tracing MPLS Label Switched Path to 10.131.161.251/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.131.1 10.131.131.2 MRU 1500 [Labels: 37 Exp: 0]
L 1 10.131.131.2 10.131.141.2 MRU 1500 [Labels: 40 Exp: 0] 0 ms, ret code 8
L 2 10.131.141.2 10.131.150.2 MRU 1504 [Labels: implicit-null Exp: 0] 0 ms, ret code 8
switch#
switch# telnet 10.131.141.2
Trying 10.131.141.2 ... Open

User Access Verification

Password:
switch> en
```

The following example shows how to force an LSP traceroute out Ethernet interface 0/0:

```
switch# traceroute mpls ipv4 10.131.159.251/32

Tracing MPLS Label Switched Path to 10.131.159.251/32, timeout is 2 seconds

Type escape sequence to abort.
  0 10.131.159.246 MRU 1500 [Labels: 19 Exp: 0]
L 1 10.131.159.245 MRU 1504 [Labels: implicit-null Exp: 0] 4 ms
! 2 10.131.159.229 20 ms

switch# traceroute mpls ipv4 10.131.159.251/32 output-interface ethernet 7/1

Tracing MPLS Label Switched Path to 10.131.159.251/32, timeout is 2 seconds

Type escape sequence to abort.
  0 10.131.191.230 MRU 1500 [Labels: 18 Exp: 0]
L 1 10.131.191.229 MRU 1504 [Labels: implicit-null Exp: 0] 0 ms
! 2 10.131.159.225 1 ms
```

Example: Pacing the Transmission of Packets

The following example shows how to pace the transmission of packets:

```
switch# ping mpls ipv4 10.5.5.5/32 interval 100

Sending 5, 100-byte MPLS Echos to 10.5.5.5/32,
     timeout is 2 seconds, send interval is 100 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/36 ms PE-802
```

Example: Interrogating the Transit Router for Its Downstream Information

The following example shows sample output when a router with two output paths is interrogated:

```
switch# ping mpls ipv4 10.161.251/32 ttl 4 repeat 1 dsmap hashkey ipv4 bitmap 16

Sending 1, 100-byte MPLS Echos to 10.131.161.251/32,
     timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
L
Echo Reply received from 10.131.131.2
  DSMAP 0, DS Router Addr 10.131.141.130, DS Intf Addr 10.131.141.130
    Depth Limit 0, MRU 1500 [Labels: 54 Exp: 0]
    Multipath Addresses:
      127.0.0.3       127.0.0.6       127.0.0.9       127.0.0.10
      127.0.0.12      127.0.0.13      127.0.0.14      127.0.0.15
      127.0.0.16

  DSMAP 1, DS Router Addr 10.131.141.2, DS Intf Addr 10.131.141.2
    Depth Limit 0, MRU 1500 [Labels: 40 Exp: 0]
    Multipath Addresses:
      127.0.0.1       127.0.0.2       127.0.0.4       127.0.0.5
      127.0.0.7       127.0.0.8       127.0.0.11

Success rate is 0 percent (0/1)
```

The multipath addresses cause a packet to transit to the router with the output label stack. The **ping mpls** command is useful for determining the number of output paths, but when the router is more than one hop away a router cannot always use those addresses to get the packet to transit through the router being interrogated. This situation exists because the change in the IP header destination address might cause

the packet to be load-balanced differently by routers between the source router and the responding router. Load balancing is affected by the source address in the IP header. The following example tests load-balancing reporting from the upstream router:

```
switch# ping mpls ipv4 10.131.161.251/32 ttl 1 repeat 1 dsmap hashkey ipv4 bitmap 8

Sending 1, 100-byte MPLS Echos to 10.131.161.251/32,
     timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
L
Echo Reply received from 10.131.131.2
  DSMAP 0, DS Router Addr 10.131.141.130, DS Intf Addr 10.131.141.130
    Depth Limit 0, MRU 1500 [Labels: 54 Exp: 0]
    Multipath Addresses:
      127.0.0.3       127.0.0.5       127.0.0.7       127.0.0.8

  DSMAP 1, DS Router Addr 10.131.141.2, DS Intf Addr 10.131.141.2
    Depth Limit 0, MRU 1500 [Labels: 40 Exp: 0]
    Multipath Addresses:
      127.0.0.1       127.0.0.2       127.0.0.4       127.0.0.6

To validate that the transit router reported the proper results, determine the Echo Reply
sender address that is two hops away and consistently check the rx label that is
advertised upstream. The following is sample output:

Success rate is 0 percent (0/1)
```

The following example shows a traceroute:

```
switch# traceroute mpls ipv4 10.131.161.251/32 destination 127.0.0.6 ttl 2

Tracing MPLS Label Switched Path to 10.131.161.251/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.131.1 10.131.131.2 MRU 1500 [Labels: 37 Exp: 0]
L 1 10.131.131.2 10.131.141.2 MRU 1500 [Labels: 40 Exp: 0] 0 ms, ret code 8
L 2 10.131.141.2 10.131.150.2 MRU 1504 [Labels: implicit-null Exp: 0] 0 ms, ret code 8
switch#
switch# telnet 10.131.141.2
Trying 10.131.141.2 ... Open

User Access Verification

Password:
switch> en
```

Example: Interrogating a Router for its DSMAP

The following example shows how to interrogate the software and hardware forwarding layer for their depth limit that needs to be returned in the DSMAP TLV:

```
switch# ping mpls ipv4 10.131.159.252/32 ttl 1 dsmap

Sending 1, 100-byte MPLS Echos to 10.131.159.252/32,
     timeout is 2 seconds, send interval is 0 msec:

Codes:
  '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
L
Echo Reply received from 10.131.191.229
  DSMAP 0, DS Router Addr 10.131.159.225, DS Intf Addr 10.131.159.225
    Depth Limit 0, MRU 1508 [Labels: 18 Exp: 0]
    Multipath Addresses:
      127.0.0.1       127.0.0.2       127.0.0.3       127.0.0.4
      127.0.0.5       127.0.0.6       127.0.0.7       127.0.0.8
      127.0.0.9       127.0.0.10     127.0.0.11     127.0.0.12
      127.0.0.13     127.0.0.14     127.0.0.15     127.0.0.16
      127.0.0.17     127.0.0.18     127.0.0.19     127.0.0.20
      127.0.0.21     127.0.0.22     127.0.0.23     127.0.0.24
      127.0.0.25     127.0.0.26     127.0.0.27     127.0.0.28
      127.0.0.29     127.0.0.30     127.0.0.31     127.0.0.32
Success rate is 0 percent (0/1)
```

Example: Requesting that a Transit Router Validate the Target FEC Stack

The following example shows how to cause a transit router to validate the target FEC stack by which an LSP to be tested is identified:

```
switch# traceroute mpls ipv4 10.5.5.5/32 flags fec

Tracing MPLS Label Switched Path to 10.5.5.5/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.2.3.2 10.2.3.3 MRU 1500 [Labels: 19 Exp: 0] L 1 10.2.3.3 10.3.4.4 MRU 1500 [Labels:
19 Exp: 0] 40 ms, ret code 8 L 2 10.3.4.4 10.4.5.5 MRU 1504 [Labels: implicit-null Exp: 0]
32 ms, ret code 8 ! 3 10.4.5.5 40 ms, ret code 3
switch# ping mpls ipv4 10.5.5.5/32
Sending 5, 100-byte MPLS Echos to 10.5.5.5/32
     timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
```

```
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
!    size 100, reply addr 10.4.5.5, return code 3
!    size 100, reply addr 10.4.5.5, return code 3
!    size 100, reply addr 10.4.5.5, return code 3
!    size 100, reply addr 10.4.5.5, return code 3
!    size 100, reply addr 10.4.5.5, return code 3

Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms
```

Example: Enabling LSP Ping to Detect LSP Breakages Caused by Untagged Interfaces

The following example shows the extra label that is added to the end of the label stack when there is explicit-null label shimming:

```
switch# traceroute mpls ipv4 10.131.159.252/32 force-explicit-null

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes:
  '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.191.252 MRU 1492 [Labels: 16/18/explicit-null Exp: 0/0/0]
L 1 10.131.191.229 MRU 1508 [Labels: 18/explicit-null Exp: 0/0] 0 ms
L 2 10.131.159.225 MRU 1508 [Labels: explicit-null Exp: 0] 0 ms
! 3 10.131.159.234 4 ms
```

The following example shows the command output when there is no explicit-null label shimming:

```
switch# traceroute mpls ipv4 10.131.159.252/32

Tracing MPLS Label Switched Path to 10.131.159.252/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.131.191.252 MRU 1496 [Labels: 16/18 Exp: 0/0]
L 1 10.131.191.229 MRU 1508 [Labels: 18 Exp: 0] 4 ms
L 2 10.131.159.225 MRU 1504 [Labels: implicit-null Exp: 0] 4 ms
! 3 10.131.159.234 4 ms
```

Additional References for MPLS LSP Ping and Traceroute

For additional information related to troubleshooting MPLS connectivity with MPLS LSP ping and traceroute, see the following sections:

Related Documents

Related Topic	Document Title
Cisco NX-OS MPLS commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco NX-OS releases, and feature sets, use the Cisco MIB Locator, found at the following URL: http://www.cisco.com/go/mibs

Feature History for MPLS LSP Ping and Traceroute

Table 34-5 lists the release history for this feature.

Table 34-5 Feature History for MPLS LSP Ping and Traceroute

Feature Name	Releases	Feature Information
MPLS LSP Ping/Traceroute for LDP/TE and LSP Ping for VCCV	6.2(2)	MPLS LSP ping/traceroute for Label Distribution Protocol and traffic engineering (LDP/TE) and LSP ping for Virtual Circuit Connectivity Verification (VCCV) provide the capabilities to monitor label switched paths (LSPs) and quickly isolate Multiprotocol Label Switching (MPLS) forwarding problems. The following command was introduced or modified: **ping mpls pseudowire**,
MPLS LSP ping and traceroute	5.2(1)	This feature was introduced.

Configuration Limits for Cisco NX-OS MPLS

The configuration limits are documented in the *Cisco Nexus 7000 Series NX-OS Verified Scalability Guide*.

IETF RFCs supported by Cisco NX-OS MPLS Features

This appendix lists the IETF RFCs supported in Cisco NX-OS for MPLS.

MPLS LDP RFCs

RFCs	Title
RFC 3036	*LDP Specification*
RFC 3037	*LDP Applicability*
RFC 3478	*Graceful Restart Mechanism for Label Distribution Protocol*
RFC 3815	*Definitions of Managed Objects for the Multiprotocol Label Switching (MPLS), Label Distribution Protocol (LDP)*
RFC 5036	*LDP Specification*
RFC 5443	*LDP IGP Synchronization*

MPLS TE RFCs

RFCs	Title
RFC 2205	*Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*
RFC 2961	*RSVP Refresh Overhead Reduction Extensions*
RFC 3209	*RSVP-TE: Extensions to RSVP for LSP Tunnels*
RFC 3630	*Traffic Engineering (TE) Extensions to OSPF Version 2*
RFC 3784	*Intermediate System to Intermediate System (IS-IS) Extensions for Traffic Engineering (TE)*
RFC 3812	*MPLS TE MIB*
RFC 4090	*Fast Reroute Extensions to RSVP-TE for LSP Tunnels*

MPLS Layer 2 VPN RFCs

RFCs	Title
RFC 2113	*IP Router Alert Option*
RFC 3032	*MPLS Label Stack Encoding*
RFC 3036	*LDP Specification*
RFC 3916	*Requirements for Pseudo-wire Emulation Edge-to-Edge (PWE3)*
RFC 3985	*Pseudo Wire Emulation Edge-to-Edge Architecture*
RFC 4379	*Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*
RFC 4447	*Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)*
RFC 4448	*Encapsulation Methods for Transport of Ethernet over MPLS Networks*
RFC 4761	*Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*
RFC 4762	*Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*
RFC 5085	*Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires*
RFC 6074	*Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)*
draft-martini-l2circuit-trans-mpls-08	*Transport of Layer 2 Frames Over MPLS*
draft-martini-l2circuit-encap-mpls-04.	*Encapsulation Methods for Transport of Layer 2 Frames Over MPLS*

MPLS Layer 3 VPN RFCs

RFCs	Title
RFC 2547	*BGP/MPLS VPNs*
RFC 2685	*Virtual Private Networks Identifier*
RFC 3107	*Carrying Label Information in BGP-4* **Note** The labeled unicast subsequent address family identifier (SAFI) value 4 that is specified in RFC 3107 applies to both IPv4 and IPv6 address family identifiers (AFIs). As of this publication date, we only support labeled unicast for IPv6 AFI.
RFC 4360	*BGP Extended Communities Attribute*
RFC 4364	*BGP/MPLS IP Virtual Private Networks (VPNs)*
RFC 4382	*MPLS/BGP Layer 3 Virtual Private Network (VPN) MIB*
RFC 4577	*OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*
RFC 4659	*BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*
RFC 4684	*Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*
RFC 4760	*Multiprotocol Extensions for BGP-4*
RFC 4781	*Graceful Restart Mechanism for BGP with MPLS*

RFCs	Title
RFC 4798	*Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)*
draft-retana-bgp-custom-decision-00	*BGP Custom Decision Process*

MPLS MVPN RFCs

RFCs	Title
draft-rosen-vpn-mcast-10	*Multicast in MPLS/BGP IP VPNs*

MPLS MVPN RFCs

RFCs	Title
RFC 2113	*IP Router Alert Option*
RFC 3443	*Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks*
RFC 4377	*Operations and Management (OAM) Requirements for Multi-Protocol Label Switched (MPLS) Networks*
RFC 4378	*A Framework for Multi-Protocol Label Switching (MPLS) Operations and Management (OAM)*
RFC 4379	*Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*