

MPLS Configuration Guide for Cisco NCS 5500 Series Routers, IOS XR Release 26.1.x, 26.2.x, 26.3.x

First Published: 2026-02-28

Last Modified: 2026-09-11

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

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

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

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

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

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

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

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

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

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

© 2026 Cisco Systems, Inc. All rights reserved.

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business results 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.

© 2026 Cisco Systems, Inc. All rights reserved.



CONTENTS

PREFACE

Preface xi

Changes to This Document xi

Communications, Services, and Additional Information xi

CHAPTER 1

New and Changed MPLS Features 1

New and Changed MPLS Feature Information 1

CHAPTER 2

YANG Data Models for MPLS Features 3

Using YANG Data Models 3

CHAPTER 3

Implementing MPLS Label Distribution Protocol 5

Implementing MPLS Label Distribution Protocol 5

Prerequisites for Implementing MPLS Label Distribution Protocol 7

Restrictions for MPLS LDP 7

Overview of Label Distribution Protocol 8

Configuring Label Distribution Protocol 8

Configuring Label Distribution Protocol 8

Configuring Label Distribution Protocol Discovery Parameters 9

Label Distribution Protocol Discovery for Targeted Hellos 10

Label Advertisement Control 11

Configuring Local Label Allocation Control 12

Configuring Downstream on Demand 12

Configuring Explicit Null Label 13

Label Distribution Protocol Auto-configuration 13

Configuring Session Protection 14

Configuring Label Distribution Protocol- Interior Gateway Protocol (IGP) Synchronization 14

Configuring Label Distribution Protocol Graceful Restart 15
Configuring Label Distribution Protocol Nonstop Routing 16
Configuring LDPv6 16
Restrictions for MPLS LDPv6 17
IPv6 Support in MPLS LDP 17
LDPv6 Function 17
Feature Support in LDPv6 20
Configuring ISIS for IPv6 and LDPv6 26
LDPv6 Configuration Examples 28
MPLS Label Distribution Protocol : Details 30
Setting Up Label Switched Paths 30
Details of Label Distribution Protocol Graceful Restart 32
Details of Session Protection 35
Controlling State Advertisements In An mLDP-Only Setup 35
Use Cases For Controlling State Advertisements 37
mLDP-Based MVPN 37
Disable Prefix-LSPs On An L2VPN/PW tLDP Session 38
MPLS traffic flow control for TTL and QoS propagation 41
Configure MPLS Push, Pop, and PHP exclude operations 44
Granular TTL propagation and DSCP preservation controls on MPLS PHP nodes 48
Configure granular TTL propagation and DSCP preservation controls on MPLS PHP nodes 49

CHAPTER 4 **Implementing MPLS Static Labeling 51**
MPLS Static Labeling 51
Define Label Range and Enable MPLS Encapsulation 52
Identify and Clear Label Discrepancy 53
Configuring MPLS Static Labels for IPv6 Prefixes 54
Configuring Backup within a Forwarding Set 55
Configuring Static LSP Next Hop Resolve 58
Configuring Static LSP Next Hop Resolve with Recursive Prefix 59
Configuring MPLS Static over BVI 60
MPLS Over Single-Pass GRE Tunnels 60
Multicast Decapsulated Flow Over Static LSP GRE Tunnels 63

CHAPTER 5 **Implementing RSVP for MPLS-TE** 65

Implementing RSVP for MPLS-TE 65

Usage Guidelines and Limitations 65

Setting up MPLS LSP Using RSVP 65

Overview of RSVP for MPLS-TE Features 66

Configuring RSVP for MPLS-TE 66

Configuring RSVP Message Authentication Globally 67

Configuring RSVP Authentication for an Interface 68

Configuring RSVP Authentication on a Neighbor 68

MPLS-TE PCE authentication 69

Configuring Graceful Restart 71

Configuring Refresh Reduction 72

Configuring ACL Based Prefix Filtering 73

Configuring RSVP Packet Dropping 73

Enabling RSVP Traps 74

RSVP for MPLS-TE Features- Details 74

CHAPTER 6 **Implementing MPLS Traffic Engineering** 81

Implementing MPLS Traffic Engineering 81

Overview of MPLS-TE Features 82

How MPLS-TE Works 83

RSVP-TE Bandwidth Accounting 84

Configuring MPLS-TE 86

Building MPLS-TE Topology 86

Creating an MPLS-TE Tunnel 87

Teardown and Reestablishment of RSVP-TE Tunnels 89

Configuring Fast Reroute 92

Configuring Auto-Tunnel Backup 93

Configuring Next Hop Backup Tunnel 96

Configuring SRLG Node Protection 96

Configuring Pre-Standard DS-TE 97

Configuring an IETF DS-TE Tunnel Using RDM 98

Configuring an IETF DS-TE Tunnel Using MAM 99

Configuring Flexible Name-Based Tunnel Constraints 99
Configuring Automatic Bandwidth 100
Enable per-application accounting for interfaces 101
Configure RSVP-TE dark bandwidth accounting 101
Autoroute Announce 104
Configuring Auto-Tunnel Mesh 113
Configuring an MPLS Traffic Engineering Interarea Tunneling 114
Configuring LDP over MPLS-TE 115
Configuring MPLS-TE Path Protection 116
Configuring Auto-bandwidth Bundle TE++ 119
Configure Autoroute Tunnel as Designated Path 122
Configure IPv6 routing over IPv4 MPLS-TE tunnels 125
Configure IPv6 routing over IPv6 MPLS-TE tunnels 127
Regular expressions in MPLS-TE tunnel names 128
Limitations of using regex in MPLS-TE tunnels 129
Usage guidelines of regex in MPLS-TE tunnel names 130
Regular expressions in MPLS-TE tunnel name filtering commands 131
MPLS-TE Features - Details 132
UCMP Over MPLS-TE 135
Event history for MPLS-TE tunnels 140
Benefits of tunnel event history 142
Restrictions for tunnel event history 142
Configure tunnel event history and event count 142
Configuring Performance Measurement 146

CHAPTER 7 **Implementing MPLS OAM** 149
Implementing MPLS OAM 149
MPLS LSP Ping 149
MPLS LSP Traceroute 151
MPLS OAM Using Nil FEC 152

CHAPTER 8 **Implementing GMPLS UNI** 155
GMPLS UNI 155
GMPLS UNI Configurations 162

Verifying GMPLS UNI Configurations 165

Additional References 174



Preface

This preface contains these sections:

- Changes to This Document , on page xi
- Communications, Services, and Additional Information, on page xi

Changes to This Document

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

Table 1: Changes to This Document

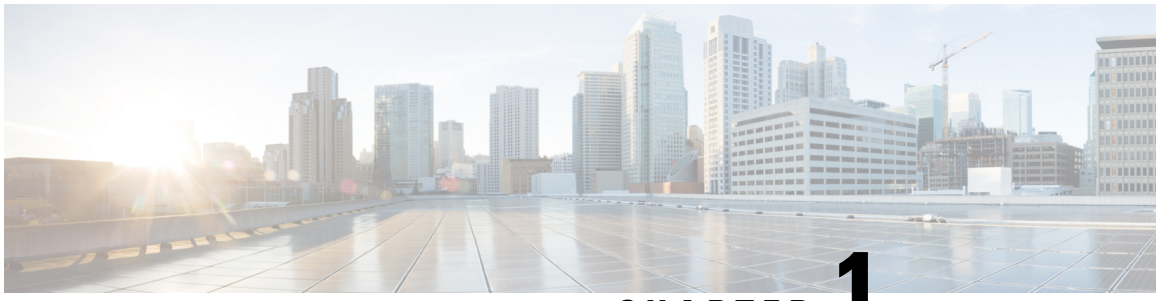
Date	Change Summary
September 2026	Republished for Release 26.3.1.
June 2026	Republished for Release 26.2.1.
February 2026	Initial release of this document.

Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business results 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.



CHAPTER 1

New and Changed MPLS Features

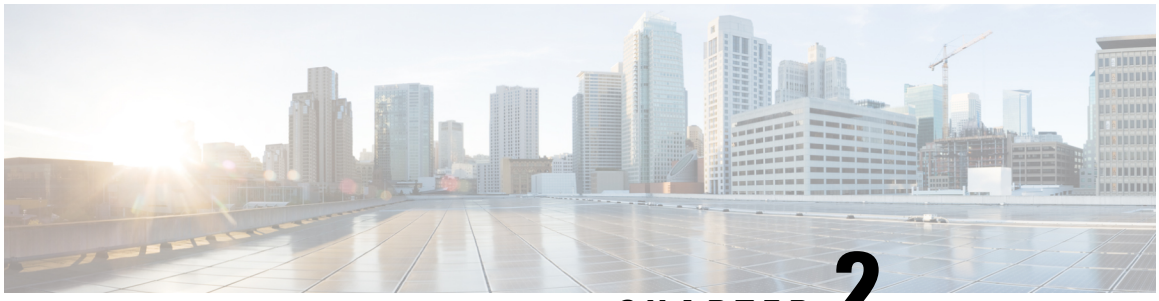
This table summarizes the new and changed feature information for the *MPLS Configuration Guide for Cisco NCS 5500 Series Routers*, and tells you where they are documented.

- New and Changed MPLS Feature Information, on page 1

New and Changed MPLS Feature Information

Table 2: New and Changed Features

Feature	Description	Changed in Release	Where Documented
Regular expressions in MPLS-TE tunnel names	This feature was introduced.	Release 26.2.1	Regular expressions in MPLS-TE tunnel names
None	No new features introduced.	Release 26.1.1	Not applicable



CHAPTER 2

YANG Data Models for MPLS Features

This chapter provides information about the YANG data models for MPLS features.

- Using YANG Data Models, on page 3

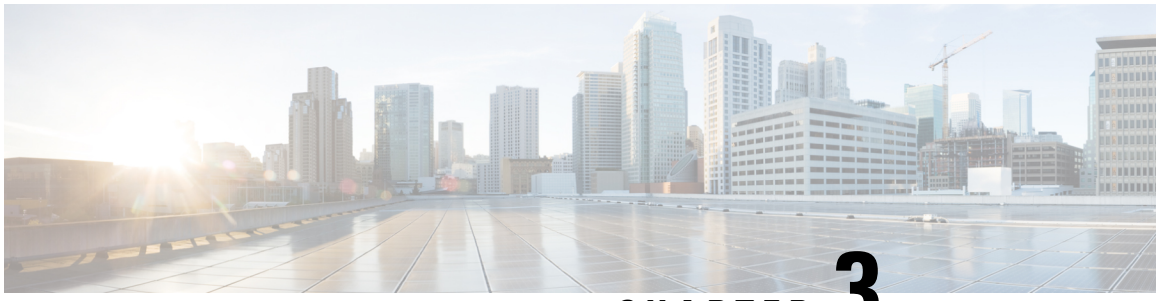
Using YANG Data Models

Cisco IOS XR supports a programmatic way of configuring and collecting operational data of a network device using YANG data models. Although configurations using CLIs are easier and human-readable, automating the configuration using model-driven programmability results in scalability.

The data models are available in the release image, and are also published in the Github repository. Navigate to the release folder of interest to view the list of supported data models and their definitions. Each data model defines a complete and cohesive model, or augments an existing data model with additional XPaths. To view a comprehensive list of the data models supported in a release, navigate to the **Available-Content.md** file in the repository.

You can also view the data model definitions using the YANG Data Models Navigator tool. This GUI-based and easy-to-use tool helps you explore the nuances of the data model and view the dependencies between various containers in the model. You can view the list of models supported across Cisco IOS XR releases and platforms, locate a specific model, view the containers and their respective lists, leaves, and leaf lists presented visually in a tree structure. This visual tree form helps you get insights into nodes that can help you automate your network.

To get started with using the data models, see the *Programmability Configuration Guide*.



CHAPTER 3

Implementing MPLS Label Distribution Protocol

- Implementing MPLS Label Distribution Protocol, on page 5
- Prerequisites for Implementing MPLS Label Distribution Protocol, on page 7
- Restrictions for MPLS LDP, on page 7
- Overview of Label Distribution Protocol, on page 8
- Configuring Label Distribution Protocol, on page 8
- Configuring LDPv6 , on page 16
- MPLS Label Distribution Protocol : Details, on page 30
- Controlling State Advertisements In An mLDP-Only Setup, on page 35
- Use Cases For Controlling State Advertisements, on page 37
- MPLS traffic flow control for TTL and QoS propagation, on page 41
- Granular TTL propagation and DSCP preservation controls on MPLS PHP nodes, on page 48

Implementing MPLS Label Distribution Protocol

MPLS (Multi Protocol Label Switching) is a forwarding mechanism based on label switching. In an MPLS network, data packets are assigned labels and packet-forwarding decisions are taken based on the contents of the label. To switch labeled packets across the MPLS network, predetermined paths are established for various source-destination pairs. These predetermined paths are known as Label Switched Paths (LSPs). To establish LSPs, MPLS signaling protocols are used. Label Distribution Protocol (LDP) is an MPLS signaling protocol used for establishing LSPs. This module provides information about how to configure MPLS LDP.

IPv6 Support in MPLS LDP-MPLS LDPv6 makes the LDP control plane to run on IPv6 in order to setup LSPs for IPv6 prefixes. You can enable MPLS LDPv6 along with existing IPv4 services. LDPv6 feature support is explained in the *IPv6 Support in MPLS LDP* section.

Note In the 7.2.1 release, LDPv6 is only supported for routers in an LSR role, and not the LDP Label Edge Router (LER) role.

Load Balancing with MPLS Traffic

Forwarding to a single destination can occur over multiple best paths, which tie for top place in routing metric calculations. Traffic flows are distributed across the multiple paths using a distribution algorithm, which you

can choose based on the traffic flow pattern. Load balancing decisions are made on ingress line card based on your selected load-balancing profile.

To allow hashing for the Label Edge Router (LER) and Label Switched Routers (LSRs) with MPLS traffic or algorithm to use the inner ethernet fields of the source MAC and destination MAC addresses, use the **hw-module profile load-balance algorithm** command with a suitable load-balancing profile.

You can choose a suitable load-balancing profile (mpls-lsr-ler or mpls-lsr-ler-optimized) based on the prominent traffic flow on the device as recommended in the following table.

Table 3: MPLS Load-Balancing Profiles for Different Traffic Flows

Traffic Flow	Description	Recommended Load-Balancing Profile
4 Label IPv6 flows (EthoMPLS4/6oIPv6)	Flows that have four to six MPLS labels, followed by IPv6 and next header	mpls-lsr-ler-optimized
IPv6 pop and lookup flows (EthoMPLS2/3oIPv6oXX) with L4 as non-TCP/UDP (for example, no next header, GRE)	Flows that have two or three MPLS labels, followed by IPv6 and either no next header or a next header as non-TCP/UDP (for example, GRE, ESP)	mpls-lsr-ler-optimized
IPv4 pop and lookup flows (EthoMPLS2/3oIPv4oL4) with L4 as TCP or UDP	Flows that have two or three MPLS labels, followed by IPv4 and next header as TCP/UDP. **Note** If you are running Cisco IOS XR 7.10.1 or earlier, GRE header is parsed along with IPv4 header. Therefore, if GRE options such as key and sequence number are used, IPv4 source IP and destination IP may point to incorrect offsets, resulting in out of order packets.	mpls-lsr-ler
IPv6 pop and lookup flows (EthoMPLS2/3oIPv6oXX) with L4 as TCP or UDP	Flows that have two or three MPLS labels, followed by IPv6 and next header as TCP/UDP	mpls-lsr-ler

Starting with Cisco IOS XR Release 7.10.2, when either of the following load-balancing profiles is enabled, GRE tunnel termination is disabled in the hardware.

- **hw-module profile load-balance algorithm mpls-lsr-ler**
- **hw-module profile load-balance algorithm mpls-lsr-ler-optimized**

As a result, if the router is operating as the tail end of a GRE tunnel, incoming GRE packets that would normally be decapsulated in hardware are instead punted to the line card (LC) CPU for software forwarding.

Because the punt path is rate-limited to protect the LC CPU, GRE tunnel termination is effectively unavailable and packets may be dropped under traffic load.

Supported platforms for **hw-module profile load-balance algorithm mpls-lsr-ler** and **hw-module profile load-balance algorithm mpls-lsr-ler-optimized** profiles:

- NCS5501
- NCS5502
- NCS55A2-MOD
- all NCS55A1 fixed chassis
- all NC55 line cards

Prerequisites for Implementing MPLS Label Distribution Protocol

The following are the prerequisites to implement MPLS LDP:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You must be running Cisco IOS XR software.
- You must install a composite mini-image and the MPLS package.
- You must activate IGP.
- We recommend to use a lower session holdtime bandwidth such as neighbors so that a session down occurs before an adjacency-down on a neighbor. Therefore, the following default values for the hello times are listed:
  - Holdtime is 15 seconds.
  - Interval is 5 seconds.

  For example, the LDP session holdtime can be configured as 30 seconds by using the **holdtime** command.

Restrictions for MPLS LDP

- When paths of different technologies are resolved over ECMP, it results in *heterogeneous* ECMP, leading to severe network traffic issues. Don’t use ECMP for any combination of the following technologies:
  - LDP.
  - BGP-LU, including services over BGP-LU loopback peering or recursive services at Level-3.
  - VPNv4.
  - 6PE and 6VPE.
  - EVPN.

- Recursive static routing.

Overview of Label Distribution Protocol

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
BFD, LACP Triggering TE FRR	Release 7.3.1	This feature is now supported on routers that have Cisco NC57 line cards installed and operate in the native mode.

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Targeted LDP	Release 7.3.1	This feature is now supported on routers that have Cisco NC57 line cards installed and operate in the native mode.

In 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 then forwards the packet to the next hop. MPLS is a forwarding mechanism in which packets are forwarded based on labels. Label Distribution Protocols assign, distribute, and install the labels in an MPLS environment. It is the set of procedures and messages by which Label Switched Routers (LSRs) establish LSPs through a network by mapping network-layer routing information directly to data-link layer switched paths. These LSPs may have an endpoint at a directly attached neighbor (comparable to IP hop-by-hop forwarding), or may have an endpoint at a network egress node, enabling switching via all intermediary nodes.

LSPs can be created statically, by RSVP traffic engineering (TE), or by LDP. LSPs created by LDP perform hop-by-hop path setup instead of an end-to-end path. LDP enables LSRs to discover their potential peer routers and to establish LDP sessions with those peers to exchange label binding information. Once label bindings are learned, the LDP is ready to set up the MPLS forwarding plane.

For more information about setting up LSPs, see MPLS Label Distribution Protocol : Details, on page 30.

Configuring Label Distribution Protocol

Depending on the requirements, LDP requires some basic configuration tasks described in the following topics:

Configuring Label Distribution Protocol

This section explains the basic LDP configuration. LDP should be enabled on all interfaces that connects the router to potential LDP peer routers. You can enable LDP on an interface by specifying the interface under mpls ldp configuration mode.

Configuration Example

This example shows how to enable LDP over an interface.

```
RP/0/RP0/CPU0:Router(config)#  mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)#  router-id 192.168.70.1
RP/0/RP0/CPU0:Router(config-ldp)#  interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-ldp-if)#  commit
```

Configuring Label Distribution Protocol Discovery Parameters

LSRs that are running LDP send hello messages on all the LDP enabled interfaces to discover each other. So, the LSR that receives the LDP hello message on an interface is aware of the presence of the LDP router on that interface. If LDP hello messages are sent and received on an interface, there's an LDP adjacency across the link between the two LSRs that are running LDP. By default, hello messages are sent every 5 seconds with a hold time of 15 seconds. If the LSR doesn't receive a discovery hello from peer before the hold time expires, the LSR removes the peer LSR from the list of discovered LDP neighbors. The LDP discovery parameters can be configured to change the default parameters.

LDP session between LSRs that aren't directly connected is known as targeted LDP session. For targeted LDP sessions, LDP uses targeted hello messages to discover the extended neighbors. By default, targeted hello messages are sent every 10 seconds with a hold time of 90 seconds.

Configuration Example

This example shows how to configure the following LDP discovery parameters:

- hello hold time
- hello interval
- targeted hello hold time
- targeted hello interval

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# router-id 192.168.70.1
RP/0/RP0/CPU0:Router(config-ldp)# discovery hello holdtime 30
RP/0/RP0/CPU0:Router(config-ldp)# discovery hello interval 10
RP/0/RP0/CPU0:Router(config-ldp)# discovery targeted-hello holdtime 120
RP/0/RP0/CPU0:Router(config-ldp)# discovery targeted-hello interval 15
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Verification

This section verifies the MPLS LDP discovery parameters configuration.

```
RP/0/RP0/CPU0:Router# show mpls ldp parameters
LDP Parameters:
Role: Active
Protocol Version: 1
Router ID: 192.168.70.1
Discovery:
Link Hellos:     Holdtime:30 sec, Interval:10 sec
Targeted Hellos: Holdtime:120 sec, Interval:15 sec
```

```
Quick-start: Enabled (by default)
Transport address:       IPv4: 192.168.70.1
```

Label Distribution Protocol Discovery for Targeted Hellos

LDP session between LSRs that aren't directly connected is known as targeted LDP session. For LDP neighbors which aren't directly connected, you should manually configure the LDP neighborship on both the routers.

Configuration Example

This example shows how to configure LDP for non-directly connected routers, Router 1, and Router 2.

```
RP/0/RP0/CPU0:Router1(config)#  mpls ldp
RP/0/RP0/CPU0:Router1(config-ldp)# router-id 192.168.70.1
RP/0/RP0/CPU0:Router2(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router2(config-ldp-af)#discoverey targeted-hello accept
RP/0/RP0/CPU0:Router1(config-ldp-af)# neighbor 172.20.10.10 targeted
RP/0/RP0/CPU0:Router1(config-ldp-af)# interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router1(config-ldp-if)# commit
```

```
RP/0/RP0/CPU0:Router2(config)#  mpls ldp
RP/0/RP0/CPU0:Router2(config-ldp)# router-id 172.20.10.10
RP/0/RP0/CPU0:Router2(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router2(config-ldp-af)#discoverey targeted-hello accept
RP/0/RP0/CPU0:Router2(config-ldp-af)# neighbor 192.168.70.1 targeted
RP/0/RP0/CPU0:Router2(config-ldp-af)# commit
```

Enhanced Targeted LDP Session Scale Values

Table 6: Feature History Table

Feature Name	Release Information	Feature Description
Enhanced Targeted LDP Session Scale Values	Release 7.6.1	With the unidimensional scale parameter value increased to 1999 on the router, you can now configure more targeted LDP sessions, which use targeted "hello" messages to discover extended neighbors in an MPLS network.

Starting from Cisco IOS XR Release 7.6.1, the targeted LDP session scale value is increased to 1999 on the router. Local Packet Transport Services (LPTS) maintains a table in Ternary content-addressable memory (TCAM), which helps in delivering packets to the intended application on the Routing Processor (RP). Whenever a targeted LDP session is established for an advertised prefix, two LDP entries are pushed into LPTS table corresponding to that prefix.

By default, there is a scale limit for each flow. You need to manually allocate maximum entries to LDP flow by deallocating entries of another flow that are not in use. For a targeted LDP session, two flow type entries are pushed into LPTS: **LDP-TCP-known** and **LDP-TCP-cfg-peer**. For the underlying LDP session with the peer, you need two more LDP entries in the hardware. As a result, you need to support a Unidimensional scale, and you can configure maximum entries of all the other flows.

Configure the maximum entries of a particular flow in hardware by using the following command:

```
lpts pifib hardware dynamic-flows location <loc> flow <flow_name> max <max_entries>
```

In total, you can configure upto 8000 entries for all the flows combined in the hardware. However, the total supported LDP (including targeted LDP and LDP) scale on routers is a maximum of 1999 sessions.

Apart from these flows, you can deallocate maximum entries allocated for other flows that are not in use and allocate them to **LDP-TCP-known** and **LDP-TCP-configured** flow types, which helps in achieving the required scale.

After you configure, use the following commands to see the configured values:

```
show lpts pifib dynamic flow statistics location <loc>
```

For more information on LPTS Flow Types, see chapter *LPTS* of *IP Addresses and Services Configuration Guide*.

Label Advertisement Control

LDP allows you to control the advertising and receiving of labels. You can control the exchange of label binding information by using label advertisement control (outbound filtering) or label acceptance control (inbound filtering).

Label Advertisement Control (Outbound Filtering)

Label Distribution Protocol advertises labels for all the prefixes to all its neighbors. When this is not desirable (for scalability and security reasons), you can configure LDP to perform outbound filtering for local label advertisement for one or more prefixes to one more peers. This feature is known as LDP outbound label filtering, or local label advertisement control. You can control the exchange of label binding information using the **mpls ldp label advertise** command. Using the optional keywords, you can advertise selective prefixes to all neighbors, advertise selective prefixes to defined neighbors, or disable label advertisement to all peers for all prefixes. Prefixes and peers advertised selectively are defined in the access list.

Configuration Example: Label Advertisement Control

This example shows how to configure outbound label advertisement control. In this example, neighbors are specified to advertise and receive label advertisements. Also an interface is specified for label advertisement.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise to 10.0.0.1:0 for pfx_ac11
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise interface TenGigE 0/0/0/5

RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Label Acceptance Control (Inbound Filtering)

LDP accepts labels (as remote bindings) for all prefixes from all peers. LDP operates in liberal label retention mode, which instructs LDP to keep remote bindings from all peers for a given prefix. For security reasons, or to conserve memory, you can override this behavior by configuring label binding acceptance for set of prefixes from a given peer. The ability to filter remote bindings for a defined set of prefixes is also referred to as LDP inbound label filtering or label acceptance control.

Configuration Example : Label Acceptance Control (Inbound Filtering)

This example shows how to configure label acceptance control. In this example, an LSR is configured to accept and retain label bindings from neighbors for prefixes defined in access list .

```
RP/0/RP0/CPU0:Router(config)#mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)#address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)#label remote accept from 192.168.1.1:0 for acl_1
RP/0/RP0/CPU0:Router(config-ldp-af)#label remote accept from 192.168.2.2:0 for acl_2
RP/0/RP0/CPU0:Router(config-ldp-af)#commit
```

Configuring Local Label Allocation Control

LDP creates label bindings for all IGP prefixes and receives label bindings for all IGP prefixes from all its peers. If an LSR receives label bindings from several peers for thousands of IGP prefixes, it consumes significant memory and CPU. In some scenarios, most of the LDP label bindings may not useful for any application and you may required to limit the allocation of local labels. This is accomplished using LDP local label allocation control, where an access list can be used to limit allocation of local labels to a set of prefixes. Limiting local label allocation provides several benefits, including reduced memory usage requirements, fewer local forwarding updates, and fewer network and peer updates.

Configuration Example

This example shows how to configure local label allocation using an IP access list to specify a set of prefixes that local labels can allocate and advertise.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local allocate for pfx_acl_1
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Configuring Downstream on Demand

By default, LDP uses downstream unsolicited mode in which label advertisements for all routes are received from all LDP peers. The downstream on demand feature adds support for downstream-on-demand mode, where the label is not advertised to a peer, unless the peer explicitly requests it. At the same time, since the peer does not automatically advertise labels, the label request is sent whenever the next-hop points out to a peer that no remote label has been assigned.

In downstream on demand configuration, an ACL is used to specify the set of peers for downstream on demand mode. For down stream on demand to be enabled, it needs to be configured on both peers of the session. If only one peer in the session has downstream-on-demand feature configured, then the session does not use downstream-on-demand mode.

Configuration Example

This example shows how to configure LDP Downstream on Demand.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# session downstream-on-demand with ACL1
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Configuring Explicit Null Label

Cisco MPLS LDP uses implicit or explicit null label as local label for routes or prefixes that terminate on the given LSR. These routes include all local, connected, and attached networks. By default, the null label is **implicit-null** that allows LDP control plane to implement penultimate hop popping (PHP) mechanism. When this is not desirable, you can configure **explicit-null** label that allows LDP control plane to implement ultimate hop popping (UHP) mechanism. You can configure explicit-null feature on the ultimate hop LSR. Access-lists can be used to specify the IP prefixes for which PHP is desired.

You can enforce implicit-null local label for a specific prefix by using the **implicit-null-override** command even if the prefix requires a non-null label to be allocated by default. For example, by default, an LSR allocates and advertises a non-null label for an IGP route. If you wish to terminate LSP for this route on penultimate hop of the LSR, you can enforce implicit-null label allocation and advertisement for this prefix using the **implicit-null-override** command.



Note If the outgoing label is *implicit-null* on the penultimate hop (of the label switched path), the outermost label is removed, and the payload forwarded. The payload is accounted as *MPLS* even if it is IP traffic, due to ASIC limitations in identifying the egress packet type correctly.

Configuration Example: Explicit Null

This example shows how to configure explicit null label.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise explict-null
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Configuration Example: Implicit Null Override

This example shows how to configure implicit null override for a set of prefixes.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise implicit-null-override for acl-1
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Label Distribution Protocol Auto-configuration

LDP auto-configuration allows you to automatically configure LDP on all interfaces for which the IGP protocol is enabled. Typically, LDP assigns and advertises labels for IGP routes and must often be enabled on all active interfaces by an IGP. During LDP manual configuration, you must define the set of interfaces under LDP which is a time-intensive procedure. LDP auto-configuration eliminates the need to specify the same list of interfaces under LDP and simplifies the configuration tasks.

Configuration Example: Enabling LDP Auto-Configuration for OSPF

This example shows how to enable LDP auto-configuration for a specified OSPF instance.

```
RP/0/RP0/CPU0:Router(config)# router ospf 190
RP/0/RP0/CPU0:Router(config-ospf)# mpls ldp auto-config
RP/0/RP0/CPU0:Router(config-ospf)# area 8
RP/0/RP0/CPU0:Router(config-ospf-ar)# interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-ospf-ar-if)# commit
```

Configuring Session Protection

When a new link or node comes up after a link failure, IP converges earlier and much faster than MPLS LDP and may result in MPLS traffic loss until the MPLS convergence. If a link flaps, the LDP session also flaps due to loss of link discovery. LDP session protection minimizes traffic loss, provides faster convergence, and protects existing LDP (link) sessions. When session protection is enabled for a peer, LDP starts sending targeted hello (directed discovery) in addition to basic discovery link hellos. When the direct link goes down, the targeted hellos can still be forwarded to the peer LSR over an alternative path as long as there is one. So, the LDP session stays up after the link goes down.

You can configure LDP session protection to automatically protect sessions with all or a given set of peers (as specified by peer-acl). When configured, LDP initiates backup targeted hellos automatically for neighbors for which primary link adjacencies already exist. These backup targeted hellos maintain LDP sessions when primary link adjacencies go down.

Configuration Example

This example shows how to configure LDP session protection for peers specified by the access control list peer-acl-1 for a maximum duration of 60 seconds.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# session protection for peer-acl-1 duration 60
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Configuring Label Distribution Protocol- Interior Gateway Protocol (IGP) Synchronization

Lack of synchronization between LDP and Interior Gateway Protocol (IGP) can cause MPLS traffic loss. Upon link up, for example, IGP can advertise and use a link before LDP convergence has occurred or, a link may continue to be used in IGP after an LDP session goes down.

LDP IGP synchronization coordinates LDP and IGP so that IGP advertises links with regular metrics only when MPLS LDP is converged on that link. LDP considers a link converged when at least one LDP session is up and running on the link for which LDP has sent its applicable label bindings and received at least one label binding from the peer. LDP communicates this information to IGP upon link up or session down events and IGP acts accordingly, depending on sync state.

LDP-IGP synchronization is supported for both OSPF and ISIS protocols and is configured under the corresponding IGP protocol configuration mode. Under certain circumstances, it might be required to delay declaration of re-synchronization to a configurable interval. LDP provides a configuration option to delay declaring synchronization up for up to 60 seconds. LDP communicates this information to IGP upon linkup or session down events.

From the 7.1.1 release, you can configure multiple MPLS-TE tunnel end points on an LER using the TLV 132 function in IS-IS. You can configure a maximum of 63 IPv4 addresses or 15 IPv6 addresses on an LER.

Configuring LDP IGP Synchronization: Open Shortest Path First (OSPF) Example

This example shows how to configure LDP-IGP synchronization for an OSPF instance. The synchronization delay is configured as 30 seconds.

```
RP/0/RP0/CPU0:Router(config)# router ospf 100
RP/0/RP0/CPU0:Router(config-ospf)# mpls ldp sync
RP/0/RP0/CPU0:Router(config-ospf)# mpls ldp igp sync delay 30
RP/0/RP0/CPU0:Router(config-ospf)# commit
```

Configuring LDP IGP Synchronization: Intermediate System to Intermediate System (IS-IS)

This example shows how to configure LDP-IGP synchronization for IS-IS.

```
RP/0/RP0/CPU0:Router(config)# router isis 100
RP/0/RP0/CPU0:Router(config-isis)#  interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:Router(config-isis-if-af)# mpls ldp sync
RP/0/RP0/CPU0:Router(config-isis-if-af)# commit
```

Configuring Label Distribution Protocol Graceful Restart

LDP Graceful Restart provides a mechanism for LDP peers to preserve the MPLS forwarding state when the LDP session goes down. Without LDP Graceful Restart, when an established session fails, the corresponding forwarding states are cleaned immediately from the restart and peer nodes. In this case, LDP forwarding has to restart from the beginning, causing a potential loss of data and connectivity. If LDP graceful restart is configured, traffic can continue to be forwarded without interruption, even when the LDP session restarts. The LDP graceful restart capability is negotiated between two peers during session initialization time. During session initialization, a router advertises its ability to perform LDP graceful restart by sending the graceful restart typed length value (TLV). This TLV contains the reconnect time and recovery time. The values of the reconnect and recovery times indicate the graceful restart capabilities supported by the router. The reconnect time is the amount of time the peer router waits for the restarting router to establish a connection. When a router discovers that a neighboring router is restarting, it waits until the end of the recovery time before attempting to reconnect. Recovery time is the amount of time that a neighboring router maintains its information about the restarting router.

Configuration Example

This example shows how to configure LDP graceful restart. In this example, the amount of time that a neighboring router maintains the forwarding state about the gracefully restarting router is specified as 180 seconds. The reconnect time is configured as 169 seconds.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)#  interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-ldp-if)# exit
RP/0/RP0/CPU0:Router(config-ldp)# graceful-restart
RP/0/RP0/CPU0:Router(config-ldp)# graceful-restart forwarding-state-holdtime 180
RP/0/RP0/CPU0:Router(config-ldp)# graceful-restart reconnect-timeout 169
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Configuring Label Distribution Protocol Nonstop Routing

LDP nonstop routing (NSR) functionality makes failures, such as Route Processor (RP) or Distributed Route Processor (DRP) fail over, invisible to routing peers with minimal to no disruption of convergence performance. By default, NSR is globally enabled on all LDP sessions except AToM.

A disruption in service may include any of these events:

- Route processor (RP) or distributed route processor (DRP) failover
- LDP process restart
- Minimum disruption restart (MDR)

Note Unlike graceful restart functionality, LDP NSR does not require protocol extensions and does not force software upgrades on other routers in the network, nor does LDP NSR require peer routers to support NSR. L2VPN configuration is not supported on NSR. Process failures of active LDP results in session loss and, as a result, NSR cannot be provided unless RP switchover is configured as a recovery action.

Configuration Example

This example shows how to configure LDP Non-Stop Routing.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# nsr
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Verification

```
RP/0/RP0/CPU0:Router# show mpls ldp nsr summary
Mon Dec  7 04:02:16.259 UTC
Sessions:
Total: 1, NSR-eligible: 1, Sync-ed: 0
(1 Ready)
```

Configuring LDPv6

The LDP configuration model is extended to introduce IPv6 as an option under the address family submodes that reside under LDP global and interface configurations. IPv6 address family is available under LDP global, LDP VRF global and interface configurations. LDPv6 is supported only under default VRF. LDPv6 should be enabled on all interfaces that connects the router to potential LDPv6 peer routers.

Note In the 7.2.1 release, LDPv6 is only supported for routers in an LSR role, and not the LDP Label Edge Router (LER) role.

Restrictions for MPLS LDPv6

MPLS LDPv6 has the following restrictions:

- IPv6 address family is supported only under default VRF.
- Implicit enabling of IPv6 address family is not allowed. It needs explicit enabling.
- It is recommended to configure a routable IPv6 discovery transport address when only LDP IPv6 is configured without explicitly specifying a router-id.

IPv6 Support in MPLS LDP

MPLS LDPv6 makes the LDP control plane to run on IPv6 in order to setup LSPs for IPv6 prefixes. This support enables most of the LDP functions supported on IPv4 to be extended to IPv6. In this context, support for native MPLS LDP over IPv6 is provided in order to seamlessly continue providing existing services while enabling new ones.

LDP associates a forwarding equivalence class (FEC) with each label switched path (LSP) it creates. The FEC associated with an LSP specifies which packets are mapped to that LSP. LDP establishes sessions with peers and exchanges FEC label bindings with them to enable creation of LSPs to carry MPLS traffic destined to IP prefixes.

As per RFC 5036, LDP base specification defines procedures and messages for exchanging bindings for IPv4 and IPv6 addresses and routing prefixes. LDPv6 related RFCs explain control plane and binding advertisement support for LDPv6.

The procedures of address bindings, label bindings, and forwarding setup are same for IPv4 and IPv6 address families in LDP. The only difference is that a different address format is used according to the IP address family. While a single-stack IP address family (IPv4-only or IPv6-only) enabled interfaces between a set of routers is the most typical deployment, scenarios for LSR interconnections using both IPv4 and IPv6 interfaces are also supported.

IPv6 support in MPLS LDP implements draft-ietf-mpls-ldp-ipv6 version12.

LDPv6 Function

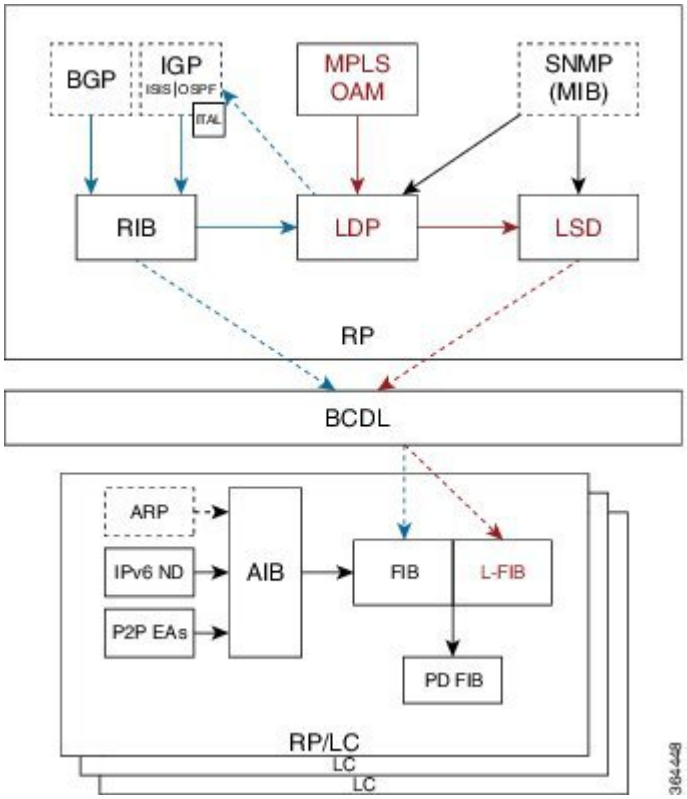
LDP functionality can be broadly divided into two categories, control plane and LSR setup.

- Control plane includes these functions - neighbor discovery (hello adjacencies), transport connection/endpoint (TCP connection), session and peering, and bindings exchange.
- LSP setup includes these functions - acquire FEC information through RIB, assign and advertise local label bindings for FEC, advertise local (interface) IP address bindings and setup forwarding rewrites.

For the control plane, the underlying address family can be either IPv4-only, IPv6-only or both. Whereas for the LSP setup, an LSP is setup for IPv4 or IPv6 FEC prefix.

This figure illustrates the main LDPv6 components.

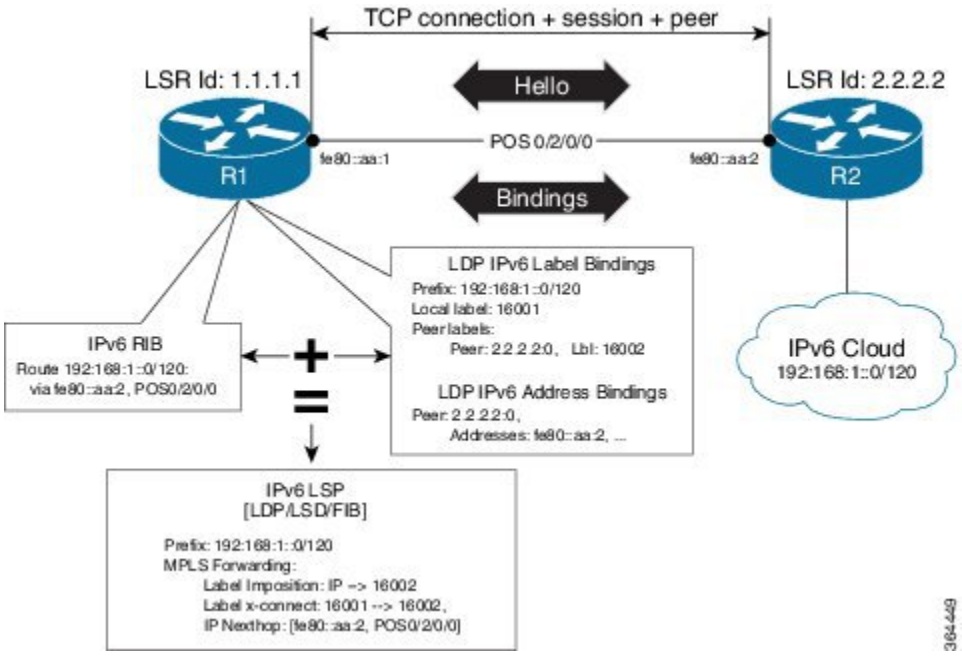
Figure 1: LDP IPv6 Architecture



LDP functions in an MPLS LDPv6 setup:

- Receive routing updates from routing information base (RIB) for global IPv6 prefixes
- Assign local labels for IPv6 prefixes
- Receive IPv6 address or state notifications for local IPv6 enabled interfaces from IP Address Repository Manager (IP-ARM/IM) and LAS for IPv6 link-local unicast addresses
- Advertise/Accept IPv6 label bindings and address bindings to/from peers
- Setup MPLS forwarding to create IPv6 LSPs
- Provide IPv6 LSP information to MPLS OAM as and when requested
- Service MIB requests for IPv6 control plane queries and generate MIB traps
- Provide LDPv6 convergence status for a link to IGP for LDP-IGP Sync feature for IPv6

Figure 2: A high level depiction of LDPv6 Control Plane and LSP Setup



Topology Scenarios

A typical deployment scenario consists of single-stack IP address-family (IPv4-only or IPv6-only) enabled interfaces between a set of routers. The following are some topology scenarios, and a description of the control plane and LSP setup scenarios.

Here, R2 is the reference router.

One dual-stack interface/same neighbor

```
R1_ _ _ _ _ _ _ _R2

IPv4+IPv6
```

- Neighbor Discovery - IPv4 and IPv6 Hellos are sent on the interface to R1.
- Transport Connection - IPv4 endpoints or IPv6 endpoints (as per user preference).
- Label binding exchange - IPv4 and IPv6 prefixes.
- Address binding exchange - IPv4 and IPv6 addresses.
- LSPs - IPv4 and IPv6 over the same nexthop interface to R1.

Two single-stack interfaces/same neighbor:

```
1. (IPv4)

R1_ _ _ _ _ _ _ _ _R2

_ _ _ _ _ _ _ _ _
2. (IPv6)
```

- Neighbor Discovery - IPv4 Hellos on interface-1 to R1, and IPv6 Hellos on interface-2 to R1.

- Transport Connection - IPv4 endpoints or IPv6 endpoints (as per user preference).
- Label binding exchange - IPv4 and IPv6 prefixes.
- Address binding exchange - IPv4 and IPv6 addresses.
- LSPs - IPv4 over nexthop interface-1 to R1, and IPv6 over nexthop interface-2 to R1.

Two single-stack interfaces/different neighbors with different address families:

```
1. (IPv4) 2. (IPv6)

R1_ _ _ _ _ _ _ _ R2_ _ _ _ _ _ _ _ R3
```

- Neighbor Discovery - IPv4 Hellos on interface-1 to R1, and IPv6 Hellos on interface-2 to R3.
- Transport Connection - IPv4 endpoints with R1 and IPv6 endpoints with R3.
- Label binding exchange - IPv4 and IPv6 prefixes to R1 and R3.
- Even if all three LSRs are dual-stack, traffic from R1 to R3 will not be completely labeled.
- If there is IPv6 traffic, it is unlabeled from R1 to R2. Labels are imposed only at R2 (although in this specific case implicit null imposition) to R3.
- If there is IPv4 traffic, it is labeled from R1 to R2. But the traffic will go unlabeled between R2 and R3 given that no IPv4 adjacency exists between R2 and R3
- Address binding exchange - IPv4 and IPv6 addresses to R1 and R3
- LSPs - IPv4 over nexthop interface-1 to R1 and IPv6 over nexthop interface-2 to R3

Feature Support in LDPv6

The following features are supported in LDPv6:

- Single-stack (native IPv6) and dual-stack (IPv4+IPv6) topologies.
- New operating modes in LDP:
  - Native LDPv6
  - LDPv6 over IPv4 and LDPv4 over IPv6 connection endpoints

    LDP Hellos carry optional transport address type length value (TLV) to notify a peer about TCP or transport connection endpoint. An LSR can include either IPv4 or IPv6 transport address TLV in an IPv4 or IPv6 Hello message. There is no difference in the TLV format of transport address for IPv4 and IPv6.

    Only one transport connection is established between two discovered peers, whether there be single address family Hello adjacencies or multi-address family (both IPv4 and IPv6) Hello adjacencies.

    In a dual-stack setup, when LDP has the option to establish transport connection either using IPv4 endpoints or IPv6 endpoints, IPv6 connection is preferred over IPv4 connection. If LDP is locally enabled for both IPv4 and IPv6 address families, every new session is treated as potential dual-stack connection. Under such circumstances, IPv6 preference is kept in place for maximum fifteen seconds for the session to establish, after which the LDP tries to establish a connection with the peer using IPv4. A user can override this default behavior by specifying the preference for a set of dual-stack peers to use IPv4 transport for the connection. Furthermore, a user may also specify maximum wait

time to wait to establish the preferred transport connection. If the preferred transport establishment times out, LDP tries to establish connection with other non-preferred transport address families. This applies to both the cases when an LSR acts as active side or passive side for the TCP connection.

To override default IPv6 transport preference for dual-stack cases, use the **mpls ldp neighbor dual-stack transport-connection prefer ipv4** command. To specify the maximum time the preferred address family connection must wait to establish a connection before resorting to a non-preferred address family, use the **mpls ldp neighbor dual-stack transport-connection max-wait** command.

Once a transport connection is established, it is not torn down depending on preferences. If the address family related to established transport connection is disabled under LDP, the corresponding transport connection is reset to reestablish the connection.

For a single-stack setup, there is no contention; the transport connection uses the given address family.

- LDP Control Plane is IPv6 aware
- LDPv6 LSP forwarding setup - LDP interacts with LSD in order to setup IPv6 LSP forwarding. The steps involved in this interaction are:
  - Label allocation for an IPv6 prefix is learnt from RIB.
  - Setup imposition and label switching forwarding path for given IPv6 prefix by creating IPv6 forwarding rewrites.
  - Like LDPv4, rewrite delete and label free operations are performed when a route disappears or is disallowed under LDP due to label policy.
  - There is no new requirement related to MPLS enabling or disabling. LDP also MPLS-enables in LSD (if not already) any LDP enabled interface, which is in the *UP* state for IP4 and/or IPv6 and has IPv4 and/or IPv6 addresses assigned.
  - In case of dual-stack LDP, a single Resource-Complete is sent by LDP to LSD once RIB-Converged notification is received for both IPv4 and IPv6 redistribute tables.
- Distribution of IPv4 and IPv6 bindings over a single LDP session established over IPv4 or IPv6
- LDP Downstream on Demand
- LDP session protection

  LDP session protection is a feature to protect an LDPv6 session. In case of dual-stack hello adjacencies with a peer, there is only a single targeted hello adjacency to protect the session. Session protection forms targeted adjacency of address family same as the transport connection. For IPv6, the target of the session protection is the remote transport connection endpoint. For IPv4, the target of the session protection is remote LSR ID.
- LDP IGPv6 sync on IPv6 interface

  This feature lets IGP support LDP IGP Sync feature for IPv6 address family. This means that Intermediate System-to-Intermediate System (IS-IS) allows IGP under an interface's IPv6 address family, whereas OSPFv3 implements it just like existing support in OSPF for IPv4. When the IGP Sync feature is enabled, LDP convergence status on an interface is considered by the IGP under the context of a given address family. This behavior applies to IGP Sync for both non-TE as well as TE tunnel interfaces.
- LDP Typed Wildcard for IPv6 prefix FEC

This feature adds support for Typed Wildcard for IPv6 Prefix FEC. The support includes:

- Being able to send or receive IPv6 Prefix Typed Wildcard FEC element in label messages.
- Respond to Typed Wildcard Label Requests received from peer by replaying its label database for IPv6 prefixes.
- Make use of Typed Wildcard Label Requests towards peers to request replay of peer label database for IPv6 prefixes. For example, on local inbound policy changes.

- Label allocation, advertisement and accept policies for IPv6 prefixes
- Local label assignment and advertisement for IPv6 default-route (::/0)
- Session MD5 authentication for IPv6 transport
- IPv6 Explicit-Null label

  IPv6 explicit null label feature support includes:

  - Advertisement and receipt of IPv6 explicit-null label to and from peers.
  - IPv6 explicit-null outgoing label in forwarding setup.
  - Explicit-null advertisement policy for a set of IPv6 prefixes and/or set of peers.
  - Explicit-null configuration change. Change in explicit-null configuration is handled by first transferring a wildcard withdraw with null label to peer(s), followed by advertising the appropriate null (implicit or explicit) label to the peer(s) again. This works without any issue as long as a single IP address family is enabled. In case of a dual-stack LSR peer, a change of configuration related to explicit-null advertisement for a given address family may cause unnecessary mix-up in the other address family.

- LDPv6 LFA FRR

  Local LFA FRR for IPv6 is supported. However, it is required that the primary and backup paths are of the same address family type, that is, an IPv4 primary path must not have an IPv6 backup path.

- NSF for LDPv6 traffic

  Non-stop forwarding (NSF) support is either provided through LDP NSR or graceful restart mechanisms.

- IGP/LDP NSR for IPv6
- IGP/LDP Graceful Restart for IPv6
- MPLS OAM: New FECs

  LSPV supports two new FECs.

  - LDPv6 Prefix FEC Encoding/Decoding

    Label Switched Path Verification (LSPV) encodes/decodes the LDP IPv6 Prefix FEC. Prefix is in the network byte order and the trailing bits are to be set to zero when prefix length is shorter than 128 bits.

  - Generic IPv6 Prefix FEC Encoding/Decoding

    LSPV encodes/decodes the generic IPv6 Prefix FEC. Prefix is in the network byte order and the trailing bits are to be set to zero when prefix length is shorter than 128 bits.

Generic IPv6 FEC is used in addition to the LDPv6 FEC. This serves the following primary purposes:

- Allows user to perform LSP ping and traceroute to verify data plane without involving control plane of the FEC in echo request and response.
- If support for a new FEC is preferred in the future, the generic FEC can be used until corresponding control plane is explicitly supported by LSPV.

- IPv6 LSR MIB

  MPLS OAM LDP MIBS is extended to support IPv6. All LSR MIB objects that reference an InSegment prefix and OutSegment next hop address are modified to support IPv6.

- LSP ping support for LDPv6
- LSP trace-route support for LDPv6
- LSP tree-trace support for LDPv6

Scale

- The same support as LDPv4 native is provided for LDPv6 native scale
- Dual-stack–The aggregate scale of LDPv4 and LDPv6 is the same as the currently support for LDPv4 native scale

Unsupported Features in LDPv6

- LDPv6 over TEv4 (traffic engineering)
- Interfaces
  - LDP auto-config for IPv6
  - LDPv6 over TEv6
  - LDPv6 over GREv6
- LDP auto-config for IPv6 IGP
- LDP Label Edge Router (LER) function
- Remote LFA FRR
- L2VPN
  - L2VPN over IPv6 LSPs
  - L2VPN signaling with LDP when the nexthop address is IPv6
- IPv6 BGP Redistribution
- Applications with native LDPv6
  - Multicast extension to LDP (mLDP) for IPv6 FEC with label binding through IPv4 and IPv6 transport
  - ICCP - ICCP and LDP ICCP with IPv6 neighbor node

- PW

- L3VPN
  - Native IPv6 MPLS L3VPNs
  - 4PE
  - 4vPE
  - LDPv6 CSC

IPv6 Label Bindings

LDP stores label bindings associated with FEC prefix in its Label Information Base (LIB) [TIB in Cisco LDP]. An entry in LIB corresponds to a prefix and holds the following bindings:

- Local binding: Local label assigned for this prefix (which is learnt through local RIB)
- Remote bindings: Array of peer labels (prefix-label bindings received in label mapping message from peers)

An entry in LIB can exist due to local binding presence, or due to remote binding(s) presence, or due to both local and remote bindings presence. The forwarding setup, however, mandates that local binding be present for a prefix.

Extensions have been implemented to support IPv6 prefixes for LIB in LDP. For per-address family convergence or preference reasons, separate or new LIB is implemented to keep and maintain IPv6 prefixes. In case of dual-stack LDP, LIBv4 is preferred over LIBv6 wherever possible. For example, during background *housekeeping* function, LIBv4 is processed before LIBv6.

IPv6 Address Bindings

LDP needs to maintain IPv6 address database for local and peer interface addresses. The IPv4 address module for local/peer addresses is extended to keep IPv4/IPv6 addresses in their respective databases, much like LIB database. In case of a dual-stack LDP, IPv4 local address database function is preferred over IPv6 local address database function where ever possible.

LDP Control Plane: Bindings Advertisement

LDP base specification allows exchange of IPv4/IPv6 bindings (address/label) on an established session. When both IPv4 and IPv6 address families are enabled under LDP, LDP distributes address/label bindings for both address families to its established peer according to local policies. Following are a few significant points pertaining to bindings support for IPv6:

- LDP allocates/advertises local label bindings for link-local IPv6 address prefixes. If received, such FEC bindings are ignored
- LDP sends only the Prefix FEC of the single address family type in a FEC TLV and not include both. If such a FEC binding is received, the entire message is ignored
- LDP sends only the addresses belonging to same address family in a single address list TLV (in address or address withdraw message)

If an address family is not enabled on receiving LSR, LDP discards any bindings received from peer(s) for the address family. This means that when address family is enabled, LDP needs to reset existing sessions with the peers in order to re-learn the discarded bindings. The implementation is optimized to reset only those sessions which were previously known to be dual-stack and had sent bindings for both address families.

LSP Mapping

LDP uses IPv6 adjacency information instead of IP address to map an IPv6 link-local nexthop to an LDP peer.

In addition to other usual checks before using a label from nexthop LDP peer, LDP uses the nexthop label for a prefix of a given address family, if there are one or more LDP hello adjacencies of the same address family type established with the peer.

Label Policies

LDP allows a user to configure label policies for allocation, acceptance, receipt, and advertisement of labels for the given prefixes.

Following are the significant points pertaining to the IPv6 support for label policies:

- Label policies and their configurations are allowed under address family IPv6
- Any policy that specifies prefix or a set of prefixes through an ACL, supports both IPv4 and IPv6 variants for address(s) or ACLs
- Any policy that specifies peer address or set of peer addresses through an ACL, supports both IPv4 and IPv6 variant for peer address(s) or ACL
- Any policy that specifies the peer's LSR ID in a peer ACL continues to take IPv4 ACL based policy irrespective of the feature configuration

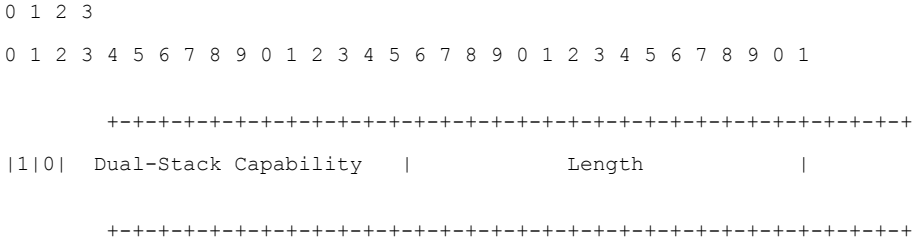
Dual-Stack Capability TLV

Clear rules are specified in RFC 5036 to determine transport connection roles in setting up a TCP connection for single-stack LDP. But RFC 5036 is not clear about dual-stack LDP, in which an LSR may assume different roles for different address families, causing issues in establishing LDP sessions.

To ensure a deterministic transport connection role for the dual-stack LDP, the dual-stack LSR conveys its transport connection preference in every LDP Hello message. This preference is encoded in a new TLV (Type Length Value) called the Dual-Stack Capability TLV. Dual-stack LSR always checks for the presence of the dual-stack capability TLV in the received LDP Hello messages and takes appropriate action for establishing or maintaining sessions.

RFC 7552 specifies more details about updates to LDPv6.

Dual-Stack Capability TLV Format



```
|   TR |     Reserved          |            MBZ               |

        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Dual-Stack Capability TLV Fields

Field	Description
U and F bits	1 and 0 (as specified by RFC 5036)
Dual-Stack Capability	TLV code point (0x0701)
TR: Transport Connection Preference	TR: Transport Connection Preference: • 0100: LDPoIPv4 connection • 0110: LDPoIPv6 connection (default)
Reserved	This field is reserved. It must be set to zero on transmission and ignore receipt
MBZ	Must be zero

Compliance Check

The compliance check prevents sessions being formed with prior RFC 7552 implementation of LDPv6.

If the dual-stack capability TLV is not present in the received Hellos and the compliance check is configured, the local and remote preferences must match to establish a session. If the preferences do not match, the LDP Hellos are dropped and the session is not established. Compliance check has therefore been disabled by default.

Use the **neighbor dual-stack tlv-compliance** command in MPLS LDP configuration to enable the compliance check.

Configuring ISIS for IPv6 and LDPv6

Intermediate System-to-Intermediate System (IS-IS) is an Interior Gateway Protocol (IGP) that advertises link-state information throughout the network to create a picture of the network topology. IPv6 IS-IS extends the address families supported by IS-IS to include IPv6, in addition to IPv4.

Previously, IS-IS supported registration of only LDP IPv4 sync status change. This has now been enhanced to support registration of notifications of LDP IPv6 sync status change. IS-IS determines the link-metrics to be advertised based on the LDP-IGP sync status on the IPv4 and IPv6 address families.

IS-IS supports non-stop forwarding (NSF) by preserving the LDPv6-IGP sync status across high availability (HA) events of IS-IS process restarts and failover.

IS-IS also supports LDPv6-IGP sync for LFA-FRR by checking the sync status of the backup interface (if it is configured with LDP IPv6 sync).

Configure Interfaces With IPv6 Addresses

```
Router# configure
Router(config)# interface Loopback 0
Router(config-if)# ipv6 address 6:6:6::6/128

Router(config)# interface HundredGigE0/0/0/0
Router(config-if)# ipv6 address 16:1::6/120
```

Enable IS-IS

Enable ISIS routing protocol and associate the IPv6 interfaces to it. Enable IPv6 capability within ISIS.

```
Router(config)# router isis 100
Router(config-isis)# net 49.0000.0000.0000.0006.00
Router(config-isis)# interface Loopback 0
Router(config-isis-if)# address-family ipv6 unicast

Router(config-isis)# interface HundredGigE0/0/0/0
Router(config-isis-if)# address-family ipv6 unicast
```

Implement LDPv6 Globally

```
Router(config)# mpls ldp
Router(config-ldp)# address-family ipv6
```

Configure Default Transport Address

LDP computes default local transport address for IPv6 from its IPv6 interface or address database by picking the lowest operational loopback interface with global unicast IPv6 address. This means that any change in this loopback state or address, flaps or changes the default transport address for IPv6 and may cause session flaps using such an address as transport endpoint. For example, if a session is currently active on Loopback2 as during it's inception it was the lowest loopback with an IPv6 address, and a lower loopback, Loopback0, is configured with an IPv6 address, the session does not flap. However, if it does flap, the next time the session is attempted, Loopback0 is used.

The session flaps when configuring discovery transport address explicitly.

Use the **discovery transport-address** command under the LDP address family submode to specify the global transport address for IPv4 or IPv6.

Enable a global transport-address for the IPv6 address family.

It is recommended to configure global transport-address for IPv6 address family to avoid a potentially unstable default transport address.

```
Router(config-ldp-af)# discovery transport-address 5:6::78
```

Implement LDPv6 For The Physical Interface

```
Router(config-ldp)# interface HundredGigE0/0/0/0
Router(config-ldp-if)# address-family ipv6
```

Disable Implicit IPv4

The LDP configuration model was changed with the introduction of explicit address family enabling under LDP (VRF) global and LDP (VRF) interfaces. However, in order to support backward compatibility, the old configuration model was still supported for default VRF. There was, however, no option to disable the implicitly enabled IPv4 address family under default VRF's global or interface level.

A new configuration **mpls ldp default-vrf implicit-ipv4 disable** is now available to the user to disable the implicitly enabled IPv4 address family for the default VRF. The new configuration provides a step towards migration to new configuration model for the default VRF that mandates enabling address family explicitly. This means that if the new option is configured, the user has to explicitly enable IPv4 address family for default VRF global and interface levels. It is recommended to migrate to this explicitly enabled IPv4 configuration model.

IPv4 is implicitly enabled under default VRF and any LDP interface under default VRF. To operate as an IPv6-only LSR, disable the IPv4 address family.

```
Router(config-ldp)# default-vrf implicit-ipv4 disable
Router(config-ldp)# router-id 5.5.5.5
```

(Optional) Transport Preference Parameters

Configure IPv4 as the preferred transport (overriding the default setting of IPv6 as preferred transport) to establish connection for a set of dual-stack peers. You can also configure the maximum time (*max-wait*, in seconds) the preferred address family connection must wait to establish the transport connection before resorting to the non-preferred address family.

```
Router(config-ldp)# neighbor dual-stack transport-connection prefer ipv4
Router(config-ldp)# neighbor dual-stack transport-connection max-wait 5
```

LDPv6 Configuration Examples

The following example shows how to enable LDP IPv6 native under LDP. The user must enable IPv6 address family under LDP submodes.

```
configure
  mpls ldp
   address-family ipv6
   !
  !
```

The following example shows how to enable LDP IPv6 control plane on an LDP interface:

```
configure
  mpls ldp
   interface HundredGigE0/0/0/0
    address-family ipv6
    !
  !
```

The following examples shows how to configure IPv6-only LSR.

IPv4 is implicitly enabled under default VRF and any LDP interfaces under default VRF. In order to operate as an IPv6-only LSR, the user must also explicitly disable IPv4 address family.

LDPv6 Configuration Without Explicit IPv6 Export Address

In this example, there is no explicit IPv6 export address. The loopback's IPv6 address is used as the export address (6:6:6::6/128).

The router ID configured in MPLS LDP is not used in anyway for export. It is used only for LDP LSR identification.

```
configure
interface Loopback0
 ipv6 address 6:6:6::6/128
!
interface GigabitEthernet0/0/0/0
 ipv6 address 16:1::6/120
!
router isis 100
 net 49.0000.0000.0000.0006.00
 interface Loopback0
  address-family ipv6 unicast
  !
 !
```

```
 interface GigabitEthernet0/0/0/0
  address-family ipv6 unicast
  !
 !
mpls ldp
 default-vrf implicit-ipv4 disable
 router-id 6.6.6.6
 address-family ipv6
 !
 interface GigabitEthernet0/0/0/0
  address-family ipv6
 !
!
```

LDPv6 Configuration With Explicit IPv6 Export Address

In this example, there is an explicit IPv6 export address. However, there is no IPv6 loopback. There is no router-id configured, but the loopback IPv4 address is used.

```
Configure
!
interface Loopback0
 ipv6 address 6:6:6::6/128
!
interface HundredGigE0/0/0/0
 ipv6 address 16:1::6/120
!
router isis 100
 net 49.0000.0000.0000.0006.00
 address-family ipv6 unicast
 !
 interface Loopback0
  address-family ipv6 unicast
  !
 !
 interface HundredGigE0/0/0/0
  address-family ipv6 unicast
  !
 !
!
mpls ldp
 default-vrf implicit-ipv4 disable
 router-id 5.5.5.5
 neighbor
  dual-stack transport-connection max-wait 5
  dual-stack transport-connection prefer ipv4
 !
 address-family ipv6
  discovery transport-address 5:6::78
 !
 interface HundredGigE0/0/0/0
  address-family ipv6
 !
!
```

Associated Commands

- address-family (IS-IS)
- default-vrf implicit-ipv4 disable
- interface (IS-IS)

- ipv6 address
- mpls ldp
- neighbor dual-stack transport-connection prefer ipv4 for-peers

MPLS Label Distribution Protocol : Details

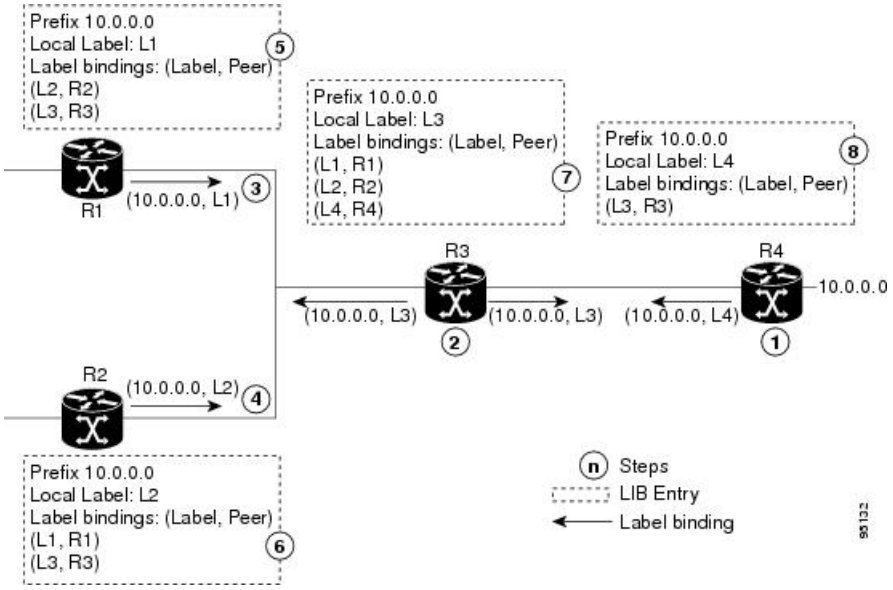
This section provides detailed conceptual information about setting up LSPs, LDP graceful restart, and LDP session protection.

Setting Up Label Switched Paths

MPLS packets are forwarded between the nodes on the MPLS network using Label Switched Paths(LSPs). LSPs can be created statically or by using a label distribution protocol like LDP. Label Switched Paths created by LDP performs hop-by-hop path setup instead of an end-to-end path. LDP enables label switched routers (LSRs) to discover their potential peer routers and to establish LDP sessions with those peers to exchange label binding information.

The following figure illustrates the process of label binding exchange for setting up LSPs.

Figure 3: Setting Up Label Switched Paths



For a given network (10.0.0.0), hop-by-hop LSPs are set up between each of the adjacent routers (or, nodes) and each node allocates a local label and passes it to its neighbor as a binding:

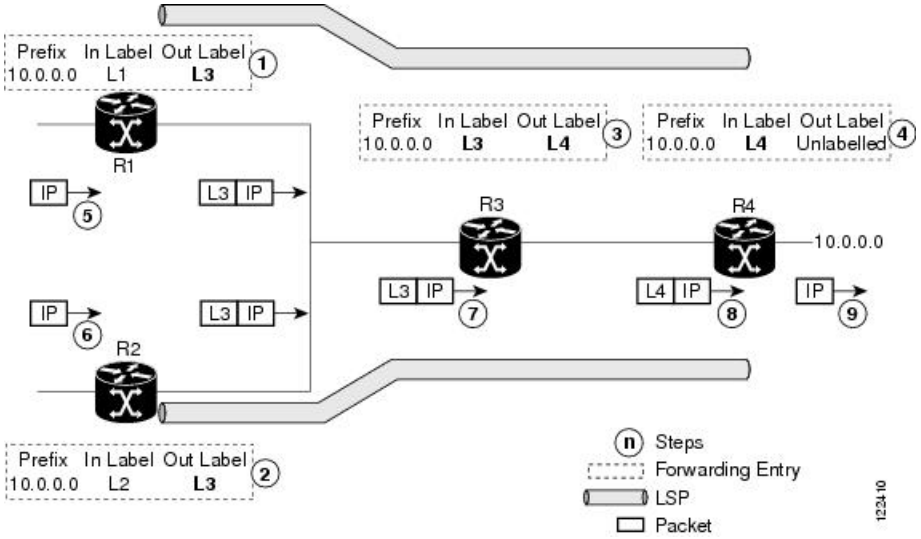
1. R4 allocates local label L4 for prefix 10.0.0.0 and advertises it to its neighbors (R3).
2. R3 allocates local label L3 for prefix 10.0.0.0 and advertises it to its neighbors (R1, R2, R4).
3. R1 allocates local label L1 for prefix 10.0.0.0 and advertises it to its neighbors (R2, R3).
4. R2 allocates local label L2 for prefix 10.0.0.0 and advertises it to its neighbors (R1, R3).

5. R1's label information base (LIB) keeps local and remote labels bindings from its neighbors.
6. R2's LIB keeps local and remote labels bindings from its neighbors.
7. R3's LIB keeps local and remote labels bindings from its neighbors.
8. R4's LIB keeps local and remote labels bindings from its neighbors.

MPLS Forwarding

Once the label bindings are learned, MPLS forwarding plane is setup and packets are forwarded as shown in the following figure.

Figure 4: MPLS Forwarding



1. Because R3 is next hop for 10.0.0.0 as notified by the FIB, R1 selects label binding from R3 and installs forwarding entry (Layer 1, Layer 3).
2. Because R3 is next hop for 10.0.0.0 (as notified by FIB), R2 selects label binding from R3 and installs forwarding entry (Layer 2, Layer 3).
3. Because R4 is next hop for 10.0.0.0 (as notified by FIB), R3 selects label binding from R4 and installs forwarding entry (Layer 3, Layer 4).
4. Because next hop for 10.0.0.0 (as notified by FIB) is beyond R4, R4 uses NO-LABEL as the outbound and installs the forwarding entry (Layer 4); the outbound packet is forwarded IP-only.
5. Incoming IP traffic on ingress LSR R1 gets label-imposed and is forwarded as an MPLS packet with label L3.
6. Incoming IP traffic on ingress LSR R2 gets label-imposed and is forwarded as an MPLS packet with label L3.
7. R3 receives an MPLS packet with label L3, looks up in the MPLS label forwarding table and switches this packet as an MPLS packet with label L4.
8. R4 receives an MPLS packet with label L4, looks up in the MPLS label forwarding table and finds that it should be Unlabeled, pops the top label, and passes it to the IP forwarding plane.

9. IP forwarding takes over and forwards the packet onward.

Details of Label Distribution Protocol Graceful Restart

LDP (Label Distribution Protocol) graceful restart provides a control plane mechanism to ensure high availability and allows detection and recovery from failure conditions while preserving Nonstop Forwarding (NSF) services. Graceful restart is a way to recover from signaling and control plane failures without impacting forwarding.

Without LDP graceful restart, when an established session fails, the corresponding forwarding states are cleaned immediately from the restarting and peer nodes. In this case LDP forwarding restarts from the beginning, causing a potential loss of data and connectivity.

The LDP graceful restart capability is negotiated between two peers during session initialization time, in FT SESSION TLV. In this typed length value (TLV), each peer advertises the following information to its peers:

Reconnect time

Advertises the maximum time that other peer will wait for this LSR to reconnect after control channel failure.

Recovery time

Advertises the maximum time that the other peer has on its side to reinstate or refresh its states with this LSR. This time is used only during session reestablishment after earlier session failure.

FT flag

Specifies whether a restart could restore the preserved (local) node state for this flag.

Once the graceful restart session parameters are conveyed and the session is up and running, graceful restart procedures are activated.

When configuring the LDP graceful restart process in a network with multiple links, targeted LDP hello adjacencies with the same neighbor, or both, make sure that graceful restart is activated on the session before any hello adjacency times out in case of neighbor control plane failures. One way of achieving this is by configuring a lower session hold time between neighbors such that session timeout occurs before hello adjacency timeout. It is recommended to set LDP session hold time using the following formula:

```
Session Holdtime <= (Hello holdtime - Hello interval) * 3
```

This means that for default values of 15 seconds and 5 seconds for link Hello holdtime and interval respectively, session hold time should be set to 30 seconds at most.

Phases in Graceful Restart

The graceful restart mechanism is divided into different phases:

Control communication failure detection

Control communication failure is detected when the system detects either:

- Missed LDP hello discovery messages
- Missed LDP keepalive protocol messages
- Detection of Transmission Control Protocol (TCP) disconnection a with a peer

Forwarding state maintenance during failure

Persistent forwarding states at each LSR are achieved through persistent storage (checkpoint) by the LDP control plane. While the control plane is in the process of recovering, the forwarding plane keeps the forwarding states, but marks them as stale. Similarly, the peer control plane also keeps (and marks as stale) the installed forwarding rewrites associated with the node that is restarting. The combination of local node forwarding and remote node forwarding plane states ensures NSF and no disruption in the traffic.

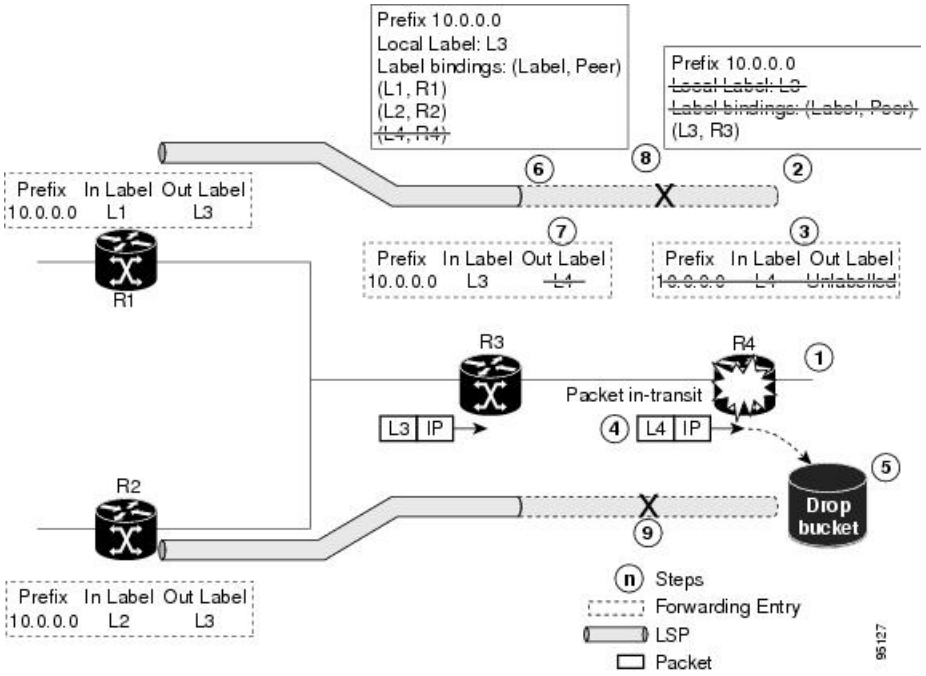
Control state recovery

Recovery occurs when the session is reestablished and label bindings are exchanged again. This process allows the peer nodes to synchronize and to refresh stale forwarding states.

Control Plane Failure

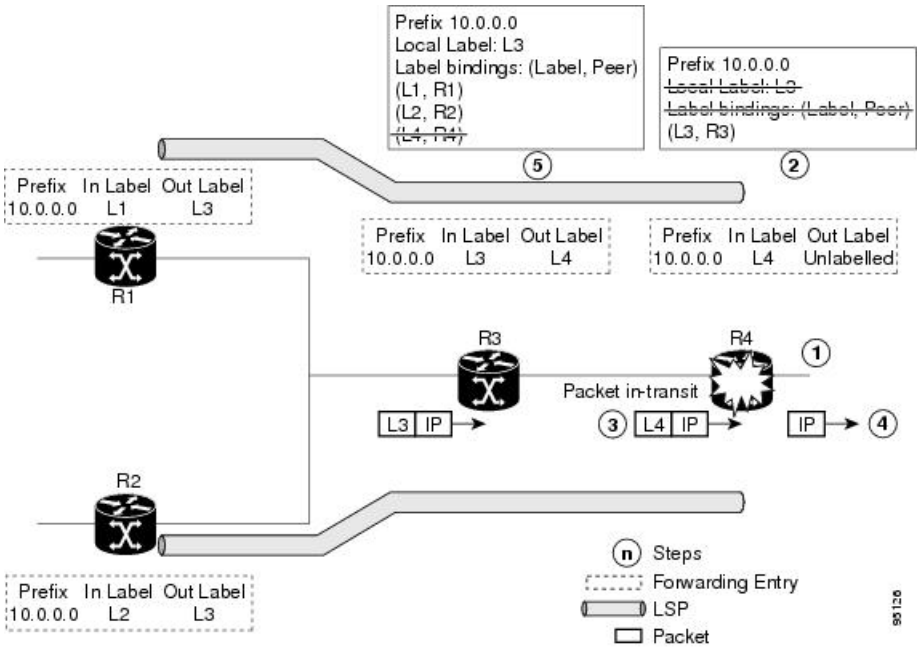
When a control plane failure occurs, connectivity can be affected. The forwarding states installed by the router control planes are lost, and the in-transit packets could be dropped, thus breaking NSF. The following figure illustrates control plane failure and recovery with graceful restart and shows the process and results of a control plane failure leading to loss of connectivity and recovery using graceful restart.

Figure 5: Control Plane Failure



Recovery with Graceful Restart

Figure 6: Recovering with Graceful Restart



1. The R4 LSR control plane restarts.
2. LIB is lost when the control plane restarts.
3. The forwarding states installed by the R4 LDP control plane are immediately deleted.
4. Any in-transit packets flowing from R3 to R4 (still labeled with L4) arrive at R4.
5. The MPLS forwarding plane at R4 performs a lookup on local label L4 which fails. Because of this failure, the packet is dropped and NSF is not met.
6. The R3 LDP peer detects the failure of the control plane channel and deletes its label bindings from R4.
7. The R3 control plane stops using outgoing labels from R4 and deletes the corresponding forwarding state (rewrites), which in turn causes forwarding disruption.
8. The established LSPs connected to R4 are terminated at R3, resulting in broken end-to-end LSPs from R1 to R4.
9. The established LSPs connected to R4 are terminated at R3, resulting in broken LSPs end-to-end from R2 to R4.

When the LDP control plane recovers, the restarting LSR starts its forwarding state hold timer and restores its forwarding state from the checkpointed data. This action reinstates the forwarding state and entries and marks them as old.

The restarting LSR reconnects to its peer, indicated in the FT Session TLV, that it either was or was not able to restore its state successfully. If it was able to restore the state, the bindings are resynchronized.

The peer LSR stops the neighbor reconnect timer (started by the restarting LSR), when the restarting peer connects and starts the neighbor recovery timer. The peer LSR checks the FT Session TLV if the restarting

peer was able to restore its state successfully. It reinstates the corresponding forwarding state entries and receives binding from the restarting peer. When the recovery timer expires, any forwarding state that is still marked as stale is deleted.

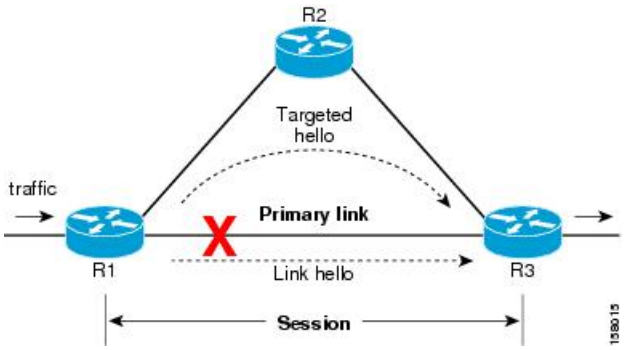
If the restarting LSR fails to recover (restart), the restarting LSR forwarding state and entries will eventually timeout and is deleted, while neighbor-related forwarding states or entries are removed by the Peer LSR on expiration of the reconnect or recovery timers.

Details of Session Protection

LDP session protection lets you configure LDP to automatically protect sessions with all or a given set of peers (as specified by peer-acl). When configured, LDP initiates backup targeted hellos automatically for neighbors for which primary link adjacencies already exist. These backup targeted hellos maintain LDP sessions when primary link adjacencies go down.

The Session Protection figure illustrates LDP session protection between neighbors R1 and R3. The primary link adjacency between R1 and R3 is directly connected link and the backup; targeted adjacency is maintained between R1 and R3. If the direct link fails, LDP link adjacency is destroyed, but the session is kept up and running using targeted hello adjacency (through R2). When the direct link comes back up, there is no change in the LDP session state and LDP can converge quickly and begin forwarding MPLS traffic.

Figure 7: Session Protection



Note When LDP session protection is activated (upon link failure), protection is maintained for an unlimited period time.

Controlling State Advertisements In An mLDP-Only Setup

This function explains the controlling of state advertisements of non-negotiated Label Distribution Protocol (LDP) applications. This implementation is in conformance with RFC 7473 (Controlling State Advertisements of Non-negotiated LDP Applications).

The main purpose of documenting this function is to use it in a Multipoint LDP (mLDP)-only environment, wherein participating routers don’t need to exchange any unicast binding information.

Non-Negotiated LDP Applications

The LDP capabilities framework enables LDP applications' capabilities exchange and negotiation, thereby enabling LSRs to send necessary LDP state. However, for the applications that existed prior to the definition of the framework (called *non-negotiated* LDP applications), there is no capability negotiation done. When an LDP session comes up, an LDP speaker may unnecessarily advertise its local state (without waiting for any capabilities exchange and negotiation). In other words, even when the peer session is established for Multipoint LDP (mLDP), the LSR advertises the state for these early LDP applications.

One example is *IPv4/IPv6 Prefix LSPs Setup* (used to set up Label Switched Paths [LSPs] for IP prefixes). Another example is *L2VPN P2P FEC 128 and FEC 129 PWs Signaling* (an LDP application that signals point-to-point [P2P] Pseudowires [PWs] for Layer 2 Virtual Private Networks [L2VPNs]).

In an mLDP-only setup, you can disable these non-negotiated LDP applications and avoid unnecessary LDP state advertisement. An LDP speaker that only runs mLDP announces to its peer(s) its disinterest (or non-support) in non-negotiated LDP applications. That is, it announces to its peers its disinterest to set up IP Prefix LSPs or to signal L2VPN P2P PW, at the time of session establishment.

Upon receipt of such a capability, the receiving LDP speaker, if supporting the capability, disables the advertisement of the state related to the application towards the sender of the capability. This new capability can also be sent later in a Capability message, either to disable a previously enabled application's state advertisement, or to enable a previously disabled application's state advertisement.

As a result, the flow of LDP state information in an mLDP-only setup is faster. When routers come up after a network event, the network convergence time is fast too.

IP Address Bindings In An mLDP Setup

An LSR typically uses peer IP address(es) to map an IP routing next hop to an LDP peer in order to implement its control plane procedures. mLDP uses a peer's IP address(es) to determine its upstream LSR to reach the root node, and to select the forwarding interface towards its downstream LSR. Hence, in an mLDP-only network, while it is desirable to disable advertisement of label bindings for IP (unicast) prefixes, disabling advertisement of IP address bindings will break mLDP functionality.

Uninteresting State - For the *Prefix-LSP* LDP application, *uninteresting* state refers to any state related to IP Prefix FEC, such as FEC label bindings and LDP Status. IP address bindings are not considered as an *uninteresting* state.

For the P2P-PW application LDP application, *uninteresting* state refers to any state related to P2P PW FEC 128 or FEC 129, such as FEC label bindings, MAC address withdrawal, and LDP PW status.

Control State Advertisement

To control advertisement of *uninteresting* state of non-negotiated LDP applications, the capability parameter TLV *State Advertisement Control Capability* is used. This TLV is only present in the Initialization and Capability messages, and the TLV can hold one or more State Advertisement Control (SAC) Elements.

As an example, consider two LSRs, S (LDP speaker) and P (LDP peer), that support all non-negotiated applications. S is participating (or set to participate) in an mLDP-only setup. Pointers for this scenario:

- By default, the LSRs will advertise state for all LDP applications to their peers, as soon as an LDP session is established.
- The **capabilities sac mldp-only** function is enabled on S.
- P receives an update from S via a Capability message that specifies to disable all four non-negotiated applications states.

- P's outbound policy towards S blocks and disables state for the unneeded applications.
- S only receives mLDP advertisements from specific mLDP-participating peers.

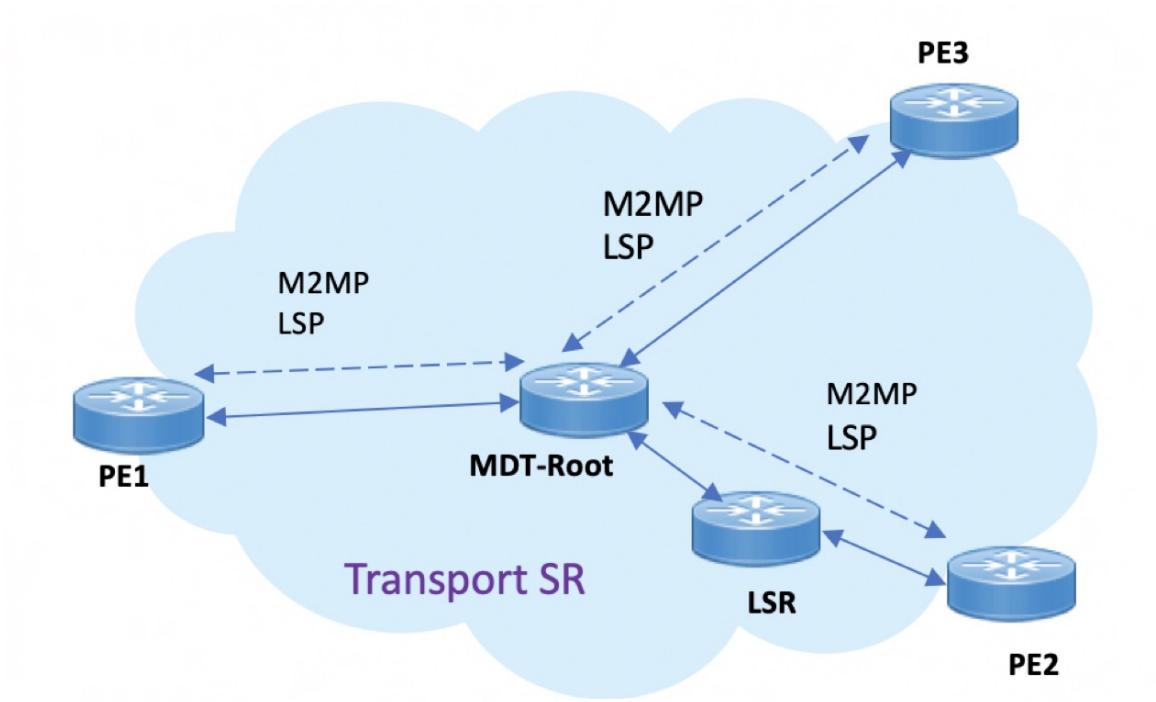
Use Cases For Controlling State Advertisements

Two use cases are explained, **mLDP-Based MVPN** and **Disable Prefix-LSPs On An L2VPN/PW tLDP Session**.

mLDP-Based MVPN

A sample topology and relevant configurations are noted below.

Figure 8: mLDP-Based MVPN Over Segment Routing



- The topology represents an MVPN profile 1 where an mLDP-based MVPN service is deployed over a Segment Routing core setup
- mLDP is required to signal MP2MP LSPs, whereas SR handles the transport.
- SAC capabilities are used to signal *mLDP-only* capability, which blocks unrequired unicast IPv4, IPv6, FEC128, and FEC129 related label binding advertisements.
- The **mldp-only** option is enabled on PE routers and P routers to remove unwanted advertisements.

Configuration

PE1 Configuration

Configure mLDP SAC capability on PE1.

```
PE1(config)# mpls ldp
PE1(config-ldp)# capabilities sac mldp-only
PE1(config-ldp)# commit
```

PE2 Configuration

Configure mLDP SAC capability on PE2.

```
PE2(config)# mpls ldp
PE2(config-ldp)# capabilities sac mldp-only
PE2(config-ldp)# commit
```

Verification

LDP peers (PE1 and PE2) are configured with **mldp-only** option, disabling all other SAC capabilities.

```
PE1# show running-config mpls ldp

mpls ldp
   capabilities sac mldp-only
   mldp
        address-family ipv4
        !

PE2# show running-config mpls ldp

mpls ldp
   capabilities sac mldp-only
   mldp
        address-family ipv4
        !
```

On PE1, verify PE2's SAC capabilities:

```
PE1# show mpls ldp neighbor 209.165.201.20 capabilities detail

Peer LDP Identifier: 209.165.201.20:0
  Capabilities:
    Sent:
      0x508  (MP: Point-to-Multipoint (P2MP))
      0x509  (MP: Multipoint-to-Multipoint (MP2MP))
      0x50b  (Typed Wildcard FEC)
      0x50d  (State Advertisement Control)
        [ {IPv4-disable}{IPv6-disable}{FEC128-disable}{FEC129-disable} ](length 4)
    Received:
      0x508  (MP: Point-to-Multipoint (P2MP))
      0x509  (MP: Multipoint-to-Multipoint (MP2MP))
      0x50b  (Typed Wildcard FEC)
      0x50d  (State Advertisement Control)
        [ {IPv4-disable}{IPv6-disable}{FEC128-disable}{FEC129-disable} ](length 4)
```

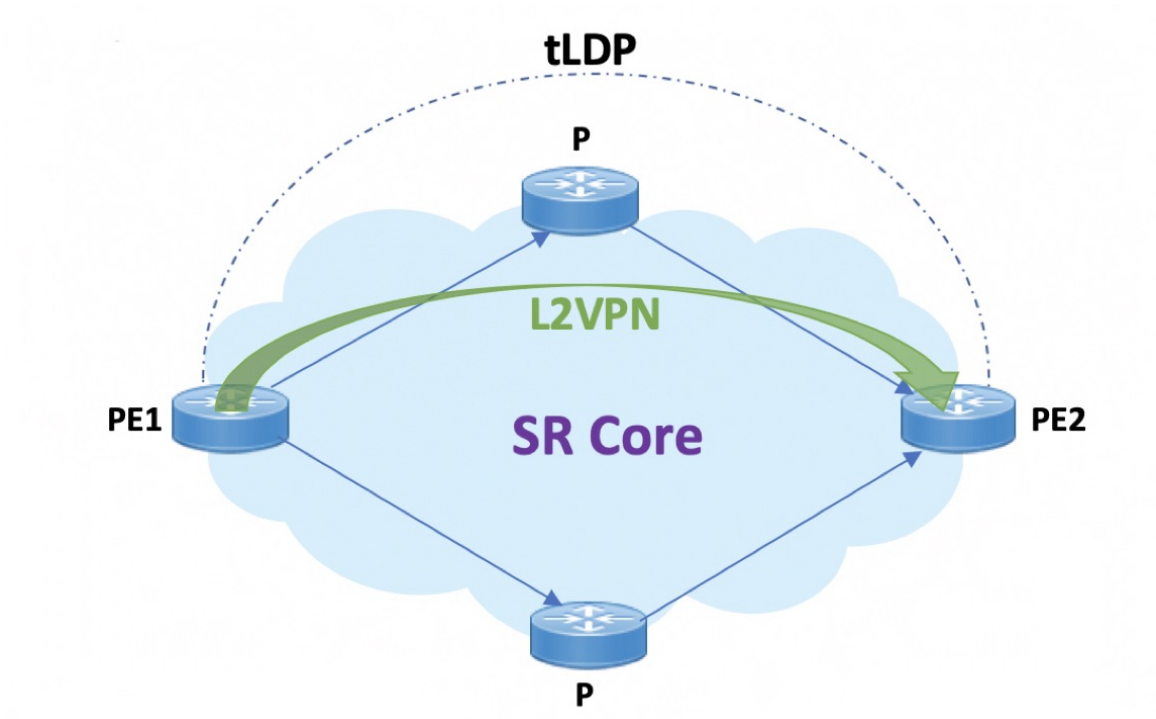
Capabilities Sent shows that **mldp-only** option disables all other advertisements.

Capabilities Received shows that **mldp-only** is enabled on peer PE2 too.

Disable Prefix-LSPs On An L2VPN/PW tLDP Session

A sample topology and relevant configurations are noted below.

Figure 9: L2VPN Xconnect Service Over Segment Routing



- The topology represents an L2VPN Xconnect service over a Segment Routing core setup.
- By default, Xconnect uses tLDP to signal service labels to remote PEs.
- By default, tLDP not only signals the service label, but also known (IPv4 and IPv6) label bindings to the tLDP peer, which is not required.
- The LDP SAC capabilities is an optional configuration enabled under LDP, and users can block IPv4 and IPv6 label bindings by applying configurations on PE1 and PE2.

Configuration

PE1 Configuration

Disable IPv4 prefix LSP binding advertisements on PE1:

```
PE1(config)# mpls ldp capabilities sac ipv4-disable
PE1(config)# commit
```

Disable IPv6-prefix LSP binding advertisements on PE1:

```
PE1(config)# mpls ldp capabilities sac ipv4-disable ipv6-disable
PE1(config)# commit
```

Note Whenever you disable a non-negotiated LDP application state on a router, you must include previously disabled non-negotiated LDP applications too, in the same command line. If not, the latest configuration overwrites the existing ones. You can see that ipv4-disable is added again, though it was already disabled.

PE2 Configuration

Enable SAC capability awareness on PE2, and make PE2 stop sending IPv4 prefix LSP binding advertisements to PE1:

```
PE2(config)#mpls ldp capabilities sac
PE2(config)#commit
```

Verification

On PE1, verify PE2's SAC capabilities:

```
PE1# show mpls ldp neighbor 198.51.100.1 detail

Peer LDP Identifier: 198.51.100.1:0
  TCP connection: 198.51.100.1:29132 - 192.0.2.1:646
  Graceful Restart: No
  Session Holdtime: 180 sec
  State: Oper; Msgs sent/rcvd: 14/14; Downstream-Unsolicited
  Up time: 00:03:30
  LDP Discovery Sources:
    IPv4: (1)
      Targeted Hello (192.0.2.1 -> 198.51.100.1, active)
    IPv6: (0)
  Addresses bound to this peer:
    IPv4: (3)
      203.0.113.1     209.165.201.1    10.0.0.1   198.51.100.1
      172.16.0.1
    IPv6: (0)
  Peer holdtime: 180 sec; KA interval: 60 sec; Peer state: Estab
  NSR: Disabled
  Clients: AToM
  Capabilities:
    Sent:
      0x508  (MP: Point-to-Multipoint (P2MP))
      0x509  (MP: Multipoint-to-Multipoint (MP2MP))
      0x50b  (Typed Wildcard FEC)
      0x50d  (State Advertisement Control)
        [ {IPv4-disable}  ] (length 1)
    Received:
      0x508  (MP: Point-to-Multipoint (P2MP))
      0x509  (MP: Multipoint-to-Multipoint (MP2MP))
      0x50b  (Typed Wildcard FEC)
      0x50d  (State Advertisement Control)
```

Capabilities Sent SAC capability **ipv4-disable** is sent, and local IPv4 label bindings are not generated.

Capabilities Received The peer (PE2) understands SAC capability and won't send its local IPv4 label bindings to local PE.

On PE1, verify SAC capabilities:

```
PE1# show mpls ldp capabilities detail

  Type     Description                                Owner
  ------   ----------------------------------------   ------------
  0x50b    Typed Wildcard FEC                         LDP
     Capability data: None

  0x3eff   Cisco IOS-XR                               LDP
     Capability data:
       Length: 12
       Desc  : [ host=PE1; platform=NCS 5500; release=07.01.01 ]
```

```
0x508    MP: Point-to-Multipoint (P2MP)             mLDP
   Capability data: None

0x509    MP: Multipoint-to-Multipoint (MP2MP)       mLDP
   Capability data: None

0x50d    State Advertisement Control                LDP
   Capability data:
     Length: 1
     Desc  : [ {IPv4-disable}  ]

0x703    P2MP PW                                    L2VPN-AToM
   Capability data: None
```

On PE1, verify that local and remote FEC bindings are removed.

```
PE1# show mpls ldp neighbor 198.51.100.1
Wed March 3 13:42:13.359 EDTs
```

MPLS traffic flow control for TTL and QoS propagation

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
MPLS traffic flow control for TTL and QoS propagation on MPLS push, pop, and penultimate nodes	Release 24.4.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]). With this feature, the extended granular control capability for the incoming and outgoing MPLS traffic changes the behavior of the IP TTL and IP QoS DSCP propagation on the MPLS push, pop, and penultimate nodes. This ensures a reduced network latency, enhanced QoS management, and simplified network operations. This feature introduces these changes: **CLI:** • **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-push ttl** • **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop ttl-and-cos** • **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl-and-cos** **YANG Data Model:** New XPaths for `Cisco-IOS-XR-um-hw-module-profile-cfg.yang` (see Github, YANG Data Models Navigator).

MPLS traffic flow control for TTL and QoS propagation on penultimate node on NC57 line cards	Release 24.4.1	Introduced in this release on: NCS 5700 fixed port routers;NCS 5700 line cards [Mode: Native]. You can now control the incoming and outgoing MPLS and IP traffic granularly to change the behavior of IP Time-to- live (TTL) and IP QoS Differentiated Services Code Point (DSCP) propagation separately on the MPLS penultimate hop popping (PHP) node using the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** and **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos** configurations within the MPLS core network. This feature ensures reduced network latency, enhanced QoS management, and simplified network operations on the MPLS penultimate nodes. This feature introduces these changes: **CLI:** • **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** • **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos** **YANG Data Model:** New XPaths for `Cisco-IOS-XR-um-hw-module-profile-cfg.yang` (see Github, YANG Data Models Navigator).

In the existing behavior, MPLS traffic flows in uniform mode by default for the IP TTL and IP QoS propagation. You can enable the MPLS traffic to flow in pipe mode using the **mpls ip-ttl-propagate disable** configuration.

Starting with Cisco IOS XR Release 24.4.1, you can control the incoming and outgoing MPLS and IP traffic for the TTL and QoS DSCP propagation so that the traffic flow in Uniform mode on the MPLS push, pop, and PHP nodes.

MPLS traffic flow control for TTL and QoS propagation on the PHP node

You can control the MPLS traffic for TTL propagation on the PHP node using the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** configuration where the TTL propagation changes to uniform mode and QoS propagation is preserved in the pipe mode.

Similarly, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos** command to control the MPLS traffic for the QoS DSCP propagation on the PHP node so that the QoS propagation changes to uniform mode whereas the TTL propagation remains in the pipe mode.

The **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** and **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos**configurations are supported only on the NC57 line cards in native mode.

To control the MPLS traffic for both the TTL and QoS DSCP propagation on the PHP node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl-and-cos** configuration so that both the IP TTL and QoS propagation changes to uniform mode.

This table provides the difference in the MPLS traffic flow behavior for the TTL propagation and QoS propagation on the PHP node between the existing **mpls ip-ttl-propagate disable** configuration and the new **hw-module fib mpls ip-ttl-propagate-disable exclude** configuration.

Table 8: Existing and new behavior for MPLS TTL and QoS propagation on the PHP node

Existing configuration	New hardware module configuration	TTL/QoS/TTL and QoS behavior modification	
mpls ip-ttl-propagate disable	**hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop {ttl | cos | ttl-and-cos}**	**Old behaviour**	**New behavior**
Not configured	Not configured	Uniform	Uniform
Not configured	Configured	Uniform	Uniform
Configured	Not configured	Pipe	Pipe
Configured	Configured	Pipe	Uniform

MPLS traffic flow control for TTL and QoS propagation on the Pop node

You can control the MPLS traffic for TTL and QoS propagation on the pop node using the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop ttl-and-cos** configuration where both the TTL propagation and QoS propagation change to uniform mode on the MPLS pop node.

This table provides the difference in the MPLS traffic flow behavior for the TTL and QoS propagation on the pop node between the existing **mpls ip-ttl-propagate disable** configuration and the new **hw-module fib mpls ip-ttl-propagate-disable exclude** configuration.

Table 9: Existing and new behavior for MPLS TTL and QoS propagation on the Pop node

Existing configuration	New hardware module cnfiguration	TTL and QoS behavior modification	
mpls ip-ttl-propagate disable	**hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop ttl-and-cos**	**Old behaviour**	**New behavior**
Not configured	Not configured	Uniform	Uniform
Not configured	Configured	Uniform	Uniform
Configured	Not configured	Pipe	Pipe
Configured	Configured	Pipe	Uniform

MPLS traffic flow control for TTL and QoS propagation on the Push node

To control the MPLS traffic for the TTL propagation on the MPLS push node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-push ttl**. This configuration changes the TTL propagation to uniform mode.

This table provides the difference in the MPLS traffic behavior for the TTL propagation on the push node between the existing **mpls ip-ttl-propagate disable** configuration and the new **hw-module fib mpls ip-ttl-propagate-disable exclude** configuration.

Table 10: Existing and new behavior for MPLS TTL propagation on the push node

Existing configuration	New hardware module configuration	TTL behavior modification	
mpls ip-ttl-propagate disable	**hw-module fib mpls ip-ttl-propagate-disable exclude mpls-push ttl**	**Old behaviour**	**New behavior**
Not configured	Not configured	Uniform	Uniform
Not configured	Configured	Uniform	Uniform
Configured	Not configured	Pipe	Pipe
Configured	Configured	Pipe	Uniform

Restrictions

- The **hw-module fib mpls ip-ttl-propagate-disable exclude** configuration works only if the **mpls ip-ttl-propagate disable** command is configured.
- The **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** and **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos** configurations are supported only on the NC57 line cards in native mode.
- The **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl-and-cos** command cannot be configured along with either the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** configuration or the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos** configuration.

Configure MPLS Push, Pop, and PHP exclude operations

You can configure the **hw-module fib mpls ip-ttl-propagate-disable exclude** command for the TTL propagation and QoS DSCP propagation on the MPLS push, pop, and penultimate hop pop (PHP) nodes.

Configure MPLS push node for TTL propagation

Configuration:

To modify the MPLS traffic flow for TTL propagation in uniform mode on the MPLS push node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-push ttl** configuration.

This example shows the configuration that changes the IP TTL propagation to uniform mode with QoS remaining in the pipe mode on the MPLS push node.

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# hw-module fib mpls ip-ttl-propagate-disable exclude mpls-push
 ttl
```

Make sure that you reload the router or all line cards on the router for the configuration to take effect.

Verification:

This example verifies whether the MPLS push node configuration for the TTL propagation is successful or not.

```
RP/0/RP0/CPU0:router# show run
hw-module fib mpls ip-ttl-propagate-disable exclude mpls-push ttl
```

This example shows whether the MPLS push node configuration for the TTL propagation is applied or not.

```
RP/0/RP0/CPU0:ios# show ofa objects global location 0/0/CPU0  | in ipttl_propagate_disable
Tue Oct 29 10:16:33.131 UTC
  ofa_bool_t ipttl_propagate_disable_mpls_push_exception => TRUE
  ofa_bool_t ipttl_propagate_disable_mpls_php_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_pop_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_cos_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_ttl_exception => FALSE
RP/0/RP0/CPU0:ios#
```

The value **TRUE** indicates that the configuration has been applied after the router reload.

Configure MPLS PHP node for TTL and Qos propagation

- **MPLS PHP node configuration for TTL propagation:**

  To modify the MPLS traffic flow for the IP TTL propagation in Uniform mode on the MPLS PHP node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl** command.

  This example shows the configuration that changes the IP TTL propagation to uniform mode with QoS remaining in the pipe mode on the MPLS PHP node.

  ```
  RP/0/RP0/CPU0:router# config
  RP/0/RP0/CPU0:router(config)# hw-module fib mpls ip-ttl-propagate-disable exclude
  mpls-pop-penultimate-hop ttl
  ```



Important Make sure that you reload the router or all line cards on the router for the configuration to take effect.

Verification:

This example verifies whether the MPLS PHP node configuration for the TTL propagation is successful or not.

```
RP/0/RP0/CPU0:router# show run
hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl
```

This example shows whether the MPLS PHP node configuration for the TTL propagation is applied or not.

```
RP/0/RP0/CPU0:ios# show ofa objects global location 0/0/CPU0  | in ipttl_propagate_disable
Tue Oct 29 10:16:33.131 UTC
  ofa_bool_t ipttl_propagate_disable_mpls_push_exception => FALSE
```

```
  ofa_bool_t ipttl_propagate_disable_mpls_php_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_pop_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_cos_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_ttl_exception => TRUE
RP/0/RP0/CPU0:ios#
```

The value **TRUE** indicates that the configuration has been applied after the router reload.

- **MPLS PHP node configuration for QoS propagation:**

  To modify the MPLS traffic flow for the QoS DSCP propagation in Uniform mode on the MPLS PHP node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos** command.

  This example shows the configuration that changes the QoS DSCP propagation to uniform mode with TTL remaining in the pipe mode on the MPLS PHP node.

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# hw-module fib mpls ip-ttl-propagate-disable exclude
mpls-pop-penultimate-hop cos
```

Important Make sure that you reload the router or all line cards on the router for the configuration to take effect.

Verification:

This example verifies whether the MPLS PHP node configuration for the QoS propagation is successful or not.

```
RP/0/RP0/CPU0:router# show run
hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop cos
```

This example shows whether the MPLS PHP node configuration for the QoS propagation is applied or not.

```
RP/0/RP0/CPU0:ios# show ofa objects global location 0/0/CPU0  | in ipttl_propagate_disable
Tue Oct 29 10:16:33.131 UTC
  ofa_bool_t ipttl_propagate_disable_mpls_push_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_pop_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_cos_exception => TRUE
  ofa_bool_t ipttl_propagate_disable_mpls_php_ttl_exception => FALSE
RP/0/RP0/CPU0:ios#
```

The value **TRUE** indicates that the configuration has been applied after the router reload.

- **MPLS PHP node configuration for TTL and QoS propagation:**

  To modify the MPLS traffic flow for both the IP TTL and QoS DSCP propagation in uniform mode on the MPLS PHP node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl-and-cos** command.

  This example shows the configuration that changes the IP TTL and QoS DSCP propagation to uniform mode on the MPLS PHP node.

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# hw-module fib mpls ip-ttl-propagate-disable exclude
mpls-pop-penultimate-hop ttl-and-cos
```

Important Make sure that you reload the router or all line cards on the router for the configuration to take effect.

Verification:

This example verifies whether the MPLS PHP node configuration for the TTL and QoS propagation is successful or not.

```
RP/0/RP0/CPU0:router# show run
hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl-and-cos
```

This example shows whether the MPLS PHP node configuration for the TTL and QoS propagation is applied or not.

```
RP/0/RP0/CPU0:ios# show ofa objects global location 0/0/CPU0  | in ipttl_propagate_disable
Tue Oct 29 10:16:33.131 UTC
  ofa_bool_t ipttl_propagate_disable_mpls_push_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_exception => TRUE
  ofa_bool_t ipttl_propagate_disable_mpls_pop_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_cos_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_ttl_exception => FALSE
RP/0/RP0/CPU0:ios#
```

The value **TRUE** indicates that the configuration has been applied after the router reload.

Configure MPLS pop node for TTL and QoS propagation

Configuration:

To modify the MPLS traffic flow for IP TTL and QoS DSCP propagation in uniform mode on the MPLS pop node, use the **hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop ttl-and-cos** configuration.

This example shows the configuration that changes both the IP TTL and QoS DSCP propagation to uniform mode on the MPLS pop node.

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop
 ttl-and-cos
```

Important Make sure that you reload the router or all line cards on the router for the configuration to take effect.

Verification:

This example verifies whether the MPLS pop node configuration for the TTL and QoS propagation is successful or not.

```
RP/0/RP0/CPU0:router# show run
hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop ttl-and-cos
```

This example shows whether the MPLS Pop node configuration for the TTL and QoS propagation is applied or not.

```
RP/0/RP0/CPU0:ios# show ofa objects global location 0/0/CPU0  | in ipttl_propagate_disable
Tue Oct 29 10:16:33.131 UTC
  ofa_bool_t ipttl_propagate_disable_mpls_push_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_pop_exception => TRUE
```

```
  ofa_bool_t ipttl_propagate_disable_mpls_php_cos_exception => FALSE
  ofa_bool_t ipttl_propagate_disable_mpls_php_ttl_exception => FALSE
RP/0/RP0/CPU0:ios#
```

The value **TRUE** indicates that the configuration has been applied after the router reload.

Granular TTL propagation and DSCP preservation controls on MPLS PHP nodes

Granular Time-to- live (TTL) propagation and Differentiated Services Code Point (DSCP) preservation controls on MPLS penultimate hop popping (PHP) nodes is a traffic engineering capability that

- allows selective propagation of TTL and IP QoS DSCP fields at the MPLS penultimate hop node,
- preserves original IP QoS DSCP values using Pipe mode, and
- inherits the TTL from the MPLS header using Uniform mode.

Table 11: Feature History Table

Feature Name	Release Information	Feature Description
Granular TTL propagation and DSCP preservation controls on MPLS PHP nodes	Release 25.3.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards). You can now customize TTL and QoS propagation independently for enhanced QoS management and simplified network operations on the MPLS penultimate nodes. This is made possible by configuring the penultimate hop node to preserve original IP QoS DSCP values using Pipe mode, while simultaneously inheriting the TTL from the MPLS header using Uniform mode. Previously, both fields followed the same mode and could not be configured separately. This feature introduces these changes: **CLI:** • **hw-module fib mpls php dscp-preserve** **YANG Data Models:** • New XPaths for `Cisco-IOS-XR-fia-hw-profile-cfg.yang` • New XPaths for `Cisco-IOS-XR-um-hw-module-profile-cfg.yang` (see Github, YANG Data Models Navigator).

Usage guidelines for configuring TTL propagation and DSCP preservation on MPLS PHP nodes

You must manually reload the chassis to activate the feature.

Restrictions for configuring TTL propagation and DSCP preservation on MPLS PHP nodes

Feature compatibility limitations

This feature cannot be configured together with

- Segment Routing SRv6 mode (**hw-module profile segment-routing srv6 mode micro-segment format**)
- MPLS extended DSCP preserve (**hw-module profile mpls-ext-dscp-preserve v4uc-enable** or **hw-module profile mpls-ext-dscp-preserve v6uc-enable**)
- MPLS IP TTL propagate disable exclusion at PHP (**hw-module fib mpls ip-ttl-propagate-disable exclude mpls-pop-penultimate-hop ttl-and-cos**)

Hardware and Platform Support

This feature is not supported on

- NCS 5700 fixed port routers, and
- NCS 5700 line cards [Mode: Compatibility; Native].

Behavioral change for TTL and QoS DSCP propagation on MPLS PHP nodes

The table shows how TTL and QoS DSCP propagation modes on MPLS PHP nodes change depending on whether the granular control feature is enabled.

Table 12: Behavioral change for TTL and QoS DSCP propagation on MPLS PHP nodes

When the feature is...	Then TTL behaves in...	And QoS behaves in...
not configured	Uniform mode	Uniform mode.
configured	Uniform mode	Pipe mode.

Configure granular TTL propagation and DSCP preservation controls on MPLS PHP nodes

Use this procedure to configure selective propagation of TTL and IP QoS DSCP fields at the MPLS PHP node.

Procedure

Step 1 Configure IP QoS DSCP value perseverance and TTL propagation.

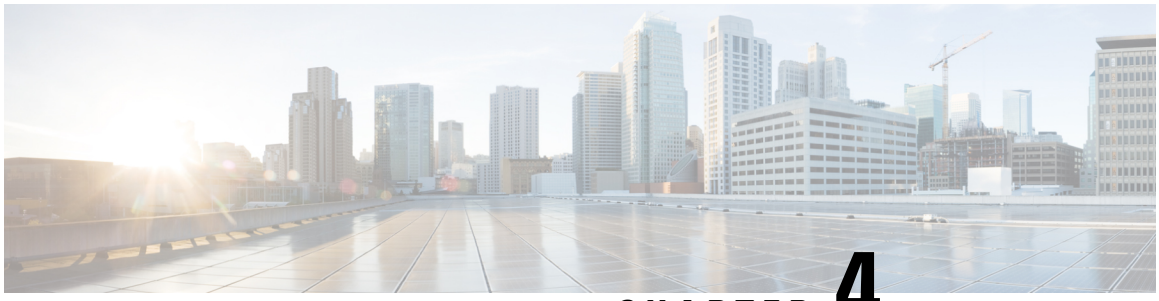
Example:

```
Router# config
```

```
Router(config)# hw-module fib mpls php dscp-preserve
Router(config)# commit
```

You must manually reload the chassis to activate the feature.

Step 2 Verify the configuration is applied to the router.



CHAPTER 4

Implementing MPLS Static Labeling

- MPLS Static Labeling , on page 51

MPLS Static Labeling

The MPLS static feature enables you to statically assign local labels to an IPv4 prefix. Also, Label Switched Paths (LSPs) can be provisioned for these static labels by specifying the next-hop information that is required to forward the packets containing static label.

If there is any discrepancy between labels assigned statically and dynamically, the router issues a warning message in the console log. By means of this warning message, the discrepancy can be identified and resolved.

The advantages of static labels over dynamic labels are:

- Improve security because the risk of receiving unwanted labels from peers (running a compromised MPLS dynamic labeling protocol) is reduced.
- Gives users full control over defined LSPs.
- Utilize system resources optimally because dynamic labeling is not processed.
- Static labeling on IPv6 packets is supported.

Restrictions

- The router does not prevent label discrepancy at the time of configuring static labels. Any generated discrepancy needs to be subsequently cleared.
- Equal-cost multi-path routing (ECMP) is not supported.
- Interfaces must be explicitly configured to handle traffic with static MPLS labels.
- The MPLS per-VRF labels cannot be shared between MPLS static and other applications.
- When paths of different technologies are resolved over ECMP, it results in *heterogeneous* ECMP, leading to severe network traffic issues. Don’t use ECMP for any combination of the following technologies:
  - LDP.
  - BGP-LU, including services over BGP-LU loopback peering or recursive services at Level-3.
  - VPNv4.

- 6PE and 6VPE.
- EVPN.
- Recursive static routing.

Define Label Range and Enable MPLS Encapsulation

By default, MPLS encapsulation is disabled on all interfaces. MPLS encapsulation has to be explicitly enabled on all ingress and egress MPLS interfaces through which the static MPLS labeled traffic travels.

Also, the dynamic label range needs to be defined. Any label that falls outside this dynamic range is available for manually allocating as static labels. The router does not verify statically-configured labels against the specified label range. Therefore, to prevent label discrepancy, ensure that you do not configure static MPLS labels that fall within the dynamic label range.



Note For Cisco IOS XR software release 7.5.2 onwards, MPLS static supports 200G Ethernet.

Configuration Example

You have to accomplish the following to complete the MPLS static labeling configuration. Values are provided as an example.

1. Define a dynamic label range, which in this task is set between 17000 and 18000.
2. Enable MPLS encapsulation on the required interface.
3. Setup a static MPLS LSP for a specific ingress label 24035.
4. Specify the forwarding information so that for packets that are received with the label, 24035, the MPLS protocol swaps labels and applies the label, 24036. After applying the new label, it forwards the packets to the next hop, 10.2.2.2, through the specified interface.

```
RP/0/RP0/CPU0:router(config)#mpls label range table 0 17000 18000
RP/0/RP0/CPU0:router(config)#commit

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

RP/0/RP0/CPU0:router(config-mpls-static)# interface HundredGigE 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-static)#address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-mpls-static-af)#local-label 24035 allocate

RP/0/RP0/CPU0:router(config-mpls-static-af-lbl)#forward
RP/0/RP0/CPU0:router(config-mpls-static-af-lbl-fwd)#
path 1 nexthop HundredGigE 0/0/0/1 10.2.2.2 out-label 24036
RP/0/RP0/CPU0:router(config-mpls-static-af-lbl-fwd)# commit
```

Verification

Verify the interfaces on which MPLS is enabled

```
RP/0/RP0/CPU0:router# show  mpls interfaces
Mon May 12 06:21:30.937 DST
Interface                  LDP      Tunnel   Static   Enabled
-------------------------- -------- -------- -------- --------
TenGigE0/0/0/5             No       No       Yes      Yes
```

Verify that the status is "Created" for the specified label value.

```
RP/0/RP0/CPU0:router#show mpls static local-label all
Tue Apr 22 18:21:55.764 UTC
Label   VRF             Type         Prefix           RW Configured   Status
------- --------------- ------------ ---------------- --------------- --------
24035   default         X-Connect    NA               Yes             Created
```

Check the dynamic range and ensure that the specified local-label value is outside this range.

```
RP/0/RP0/CPU0:router#show mpls label range
Mon Apr 28 19:56:00.596 IST
Range for dynamic labels: Min/Max: 17000/18000
```

Verify that the MPLS static configuration has taken effect, and the label forwarding is taking place.

```
RP/0/RP0/CPU0:router#show mpls lsd forwarding
Wed Nov 25 21:40:57.918 UTC
In_Label, (ID), Path_Info: <Type>
24035, (Static), 1 Paths
   1/1: IPv4, 'default':4U, BE1.2, nh=10.20.3.1, lbl=35001, flags=0x0, ext_flags=0x0
```

Associated Commands

- mpls static
- mpls label range
- show mpls interfaces

Identify and Clear Label Discrepancy

During configuring or de-configuring static labels or a label range, a label discrepancy can get generated when:

- A static label is configured for an IP prefix that already has a binding with a dynamic label.
- A static label is configured for an IP prefix, when the same label value is dynamically allocated to another IP prefix.

Verification

Identify label discrepancy by using these show commands.

```
Router#show mpls static local-label discrepancy
Tue Apr 22 18:36:31.614 UTC
Label   VRF             Type         Prefix           RW Configured   Status
------- --------------- ------------ ---------------- --------------- --------
24000   default         X-Connect     NA              Yes             Discrepancy


Router#show mpls static local-label all
Tue Apr 22 18:36:31.614 UTC
Label   VRF             Type         Prefix           RW Configured   Status
```

```
------- --------------- ------------ ---------------- --------------- --------
24000   default         X-Connect    N/A              Yes             Discrepancy
24035   default         X-Connect    N/A              Yes             Created


RP/0/RP0/CPU0:router#show log
Thu Apr 24 14:18:57.655 UTC
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level warnings, 199 messages logged
    Monitor logging: level debugging, 0 messages logged
    Trap logging: level informational, 0 messages logged
    Buffer logging: level debugging, 2 messages logged

Log Buffer (307200 bytes):

RP/0/RSP0/CPU0:Apr 24 14:18:53.743 : mpls_static[1043]:
%ROUTING-MPLS_STATIC-7-ERR_STATIC_LABEL_DISCREPANCY :
The system detected 1 label discrepancies (static label could not be allocated due to
conflict with other applications).
Please use 'clear mpls static local-label discrepancy' to fix this issue.
RP/0/RSP0/CPU0:Apr 24 14:18:53.937 : config[65762]: %MGBL-CONFIG-6-DB_COMMIT : Configuration
 committed by user 'cisco'.
Use 'show configuration commit changes 1000000020' to view the changes.
```

Rectification

Label discrepancy is cleared by allocating a new label to those IP prefixes that are allocated dynamic label. The static label configuration takes precedence while clearing discrepancy. Clearing label discrepancy may result in traffic loss for the dynamic label which got cleared.

```
Router# clear mpls static local-label discrepancy all
```

Verify that the discrepancy is cleared.

```
Router# show mpls static local-label all
Wed Nov 25 21:45:50.368 UTC
Label   VRF             Type         Prefix           RW Configured   Status
------- --------------- ------------ ---------------- --------------- --------
24000   default         X-Connect    N/A              Yes             Created
24035   default         X-Connect    N/A              Yes             Created
```

Associated Commands

- show mpls static local-label discrepancy
- clear mpls static local-label discrepancy all

Configuring MPLS Static Labels for IPv6 Prefixes

MPLS static LSPs are created by assigning static labels to incoming IPv4/IPv6 prefixes and then specifying the next hop address for the packets with statically assigned labels. Packets are forwarded in the LSP based on this information. Effective with Cisco IOS-XR 6.2.2, you can assign static labels to IPv6 prefixes and set the next hop address in the static LSP as an IPv6 address. To configure static labels for IPv6 prefixes, you first need to configure named static LSPs and perform the remaining steps under LSP configuration mode.

Configuration Example

This example shows how to assign a static label to an IPv6 prefix and also configures an IPv6 address as the next hop.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# lsp ipv6-1
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 25000 allocate per-prefix 2001:DB8:0:1::
 /64
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 nexthop TenGigabitEthernet0/0/0/0
2001:DB8:0:5::64  out-label pop
```

Verification

The following examples show how to verify MPLS static labels for IPv6 prefixes.

```
RP/0/0/CPU0:Router# show mpls forwarding

Wed March 29 12:42:41.290 IST
Local  Outgoing    Prefix              Outgoing          Next Hop            Bytes
Label  Label       or ID               Interface                             Switched
------ ----------- ------------------  ------------     ---------------      ------------
25000  Pop         2001:DB8:0:1:: /64  TenGigE0/0/0/0     2001:DB8:0:5::64    0

RP/0/0/CPU0:Router# show mpls static lsp ipv6-1 detail

Wed March 29 12:45:55.574 IST

LSP Name  Label   VRF      AFI  Type       Prefix            RW Configured   Status
--------  ------- -------  ---- ---------- ----------------  --------------- --------
ipv6-1    25000   default  IPv6 Per-Prefix 2001:DB8:0:1:: /64      Yes       Created
    PRIMARY-PATHS:
        PATH 1 :  next-hop TenGigE0/0/0/0  2001:DB8:0:5::64  out-label POP
            STATUS: VALID
```

Configuring Backup within a Forwarding Set

Various types of FRR backups can be configured between links with in a forwarding path set. You can configure the following types of FRR backups:

- Pure FRR Backup
- Reciprocal FRR backup
- One-way FRR backup

In pure FRR backup, there will be separate primary paths and backup paths. In reciprocal FRR backup, each path can act as both primary and backup. In one-way FRR backup, some paths act as both primary and backup while other paths may be just primary paths or backup paths.

Configuration Example: Pure FRR Backup

This example shows how to configure pure FRR backup with in a forwarding path set.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# lsp lsp1
```

```
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 25000 allocate
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 nexthop GigabitEthernet0/0/0/0
10.1.0.1 out-label 25000 backup-id 2
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 2 nexthop GigabitEthernet0/0/0/1
10.1.0.3 out-label 25001 backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# exit
RP/0/0/CPU0:Router(config-mpls-static-lsp)# backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# path 1 nexthop GigabitEthernet0/0/0/5
10.5.0.1 out-label pop backup-id 2
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# path 2 nexthop GigabitEthernet0/0/0/6
10.6.0.2 out-label pop backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# exit
```

The following table describes the forwarding behavior for pure FRR backup. Here P1-F and P2-F are the forwarding paths and P1-B and P2-B are the backup paths.

Action	Transient State	Interface Steady State	Forward Steady State
N/A	N/A	• P1-F: Up P2-F: Up • P1-B: Up P2-B: Up	• P1-F: Flow P2-F: Backup • P1-B: N/A P2-B: N/A
P1-F Down	P1-F FRR to P2-F	• P1-F: Up P2-F: Down • P1-B: Up P2-B: Up	• P1-F: Down P2-F: Flow • P1-B: Backup P2-B: N/A
P2-F Down	P2-F FRR to P1-B	• P1-F: Down P2-F: Down • P1-B: Up P2-B: Up	• P1-F: Down P2-F: Down • P1-B: Flow P2-B: Backup
P1-B Down	P1-B FRR to P2-B	• P1-F: Down P2-F: Down • P1-B: Down P2-B: Up	• P1-F: Down P2-F: Down • P1-B: Down P2-B: Flow

Configuration Example: Reciprocal FRR Backup

This example shows how to configure reciprocal FRR backup with in a forwarding path set.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# lsp lsp1
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 25000 allocate
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 nexthop GigabitEthernet0/0/0/0
10.1.0.1 out-label 25000 primary-and-backup backup-id 2
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 2 nexthop GigabitEthernet0/0/0/1
10.1.0.3 out-label 25001 primary-and-backup backup-id 1
```

```
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# exit
RP/0/0/CPU0:Router(config-mpls-static-lsp)# backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# path 1 nexthop GigabitEthernet0/0/0/5
10.5.0.1 out-label pop primary-and-backup backup-id 2
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# path 2 nexthop GigabitEthernet0/0/0/6
10.6.0.2 out-label pop primary-and-backup backup-id 1
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# exit
```

The following table describes the forwarding behavior for reciprocal FRR backup.

Action	Transient State	Interface Steady State	Forward Steady State
N/A	N/A	• P1-F: Up P2-F: Up • P1-B: Up P2-B: Up	• P1-F: Flow P2-F: Flow • P1-B: N/A P2-B: N/A
P2-F Down	P2-F FRR to P1-F	• P1-F: Down P2-F: Up • P1-B: Up P2-B: Up	• P1-F: Flow P2-F: Down • P1-B: Backup P2-B: N/A
P1-F Down	P1-F FRR to P1-B	• P1-F: Down P2-F: Down • P1-B: Up P2-B: Up	• P1-F: Down P2-F: Down • P1-B: Flow P2-B: Flow
P2-B Down	P2-B FRR to P1-B	• P1-F: Down P2-F: Down • P1-B: Up P2-B: Down	• P1-F: Down P2-F: Down • P1-B: Flow P2-B: Down

Configuration Example: One-way FRR Backup

This example shows how to configure one-way FRR backup with in a forwarding path set.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# lsp lsp1
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 25000 allocate
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 nexthop GigabitEthernet0/0/0/0
10.1.0.1 out-label 25000 backup-id 2
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 2 nexthop GigabitEthernet0/0/0/1
10.1.0.3 out-label 25001 primary-and-backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# exit
RP/0/0/CPU0:Router(config-mpls-static-lsp)# backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# path 1 nexthop GigabitEthernet0/0/0/5
10.5.0.1 out-label pop backup-id 2
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# path 2 nexthop GigabitEthernet0/0/0/6
10.6.0.2 out-label pop primary-and-backup
RP/0/0/CPU0:Router(config-mpls-static-lsp-backup)# exit
```

The following table describes the forwarding behavior for one-way FRR backup.

Action	Transient State	Interface Steady State	Forward Steady State
N/A	N/A	• P1-F: Up P2-F: Up • P1-B: Up P2-B: Up	• P1-F: Flow P2-F: Flow • P1-B: N/A P2-B: N/A
P2-F Down	P2-F NO-FRR to P1-F	• P1-F: Down P2-F: Up • P1-B: Up P2-B: Up	• P1-F: Flow P2-F: Down • P1-B: Backup P2-B: N/A
P1-F Down	P1-F FRR to P1-B	• P1-F: Down P2-F: Down • P1-B: Up P2-B: Up	• P1-F: Down P2-F: Down • P1-B: Flow P2-B: Flow
P1-B Down	P1-B FRR to P2-B	• P1-F: Down P2-F: Down • P1-B: Down P2-B: Up	• P1-F: Down P2-F: Down • P1-B: Down P2-B: Flow

Configuring Static LSP Next Hop Resolve

You can specify the outgoing next hop instead of explicitly specifying the outgoing path while configuring static LSPs. This next hop is resolved using the routing information base (RIB) which provides a list of paths to auto-configure. While specifying the next hop for the incoming label in a static LSP, you can specify the next hop address with out the interface using the **resolve-nexthop** command.

The following restrictions apply for this feature:

- Only supports a single next hop address which may resolve to multiple paths.
- Non-default VRFs are not supported.
- In Cisco IOS-XR 6.2.2, only host routes can be resolved. Host routes are routes with /32 mask for IPv4 addresses and /128 mask for IPv6 addresses.

Configuration Example

This example shows how to configure the static LSP next hop without specifying the interface using the **resolve-nexthop** command.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# lsp ipv6-2
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 25000 allocate per-prefix 2001:DB8:0:1::
```

```
 /64  or  24:24:1::/64
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 resolve-nexthop 2001:DB8:0:2::64
out-label pop
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# exit
```

Configuring Static LSP Next Hop Resolve with Recursive Prefix

When a routing table entry references to another IP address and not to a directly connected exit interface, the next-hop IP address is resolved using another route with an exit interface. This is known as a recursive look up because multiple lookups are required to resolve the next-hop IP address. Static LSP next hop resolve with recursive prefix feature supports resolution of recursive routes for static LSPs. In this feature, you can specify a next hop which is not directly connected using the **resolve-nexthop** command for a static LSP.

Restrictions

The following restrictions apply for this feature:

- Only eBGP routes are supported.

Configuration Example

This example shows how to configure the static LSP next hop resolve with recursive prefix. Here 192.168.2.1 is a recursive route learnt through eBGP.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# lsp anycast_5001
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 5001 allocate
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 resolve-nexthop 192.168.2.1 out-label
 pop
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# exit
```

Verification

This example shows how to verify the static LSP next hop resolve with recursive prefix configuration.

```
RP/0/0/CPU0:Router# show  mpls static lsp anycast_5001 detail
Tue Sep 12 20:00:09.248 UTC
LSP Name            Label   VRF             AFI  Type         Prefix           RW Configured
   Status
------------------- ------- --------------- ---- ------------ -----------------
--------------- --------
anycast_5001         5001    default         N/A  X-Connect    N/A              Yes
      Created
  PRIMARY SET:
    [resolve-mode: nexthop 192.168.2.1]
    Path 0 : nexthop BVI1 10.1.1.3, out-label Pop, Role: primary, Path-id: 0, Status: valid

    Path 1 : nexthop BVI1 10.1.1.4, out-label Pop, Role: primary, Path-id: 0, Status: valid

    Path 2 : nexthop BVI1 10.1.1.5, out-label Pop, Role: primary, Path-id: 0, Status: valid

    Path 3 : nexthop BVI1 10.1.1.6, out-label Pop, Role: primary, Path-id: 0, Status: valid
```

Configuring MPLS Static over BVI

A Bridge-group virtual interface (BVI) is a routed interface that represents a set of interfaces that gets bridged. By using BVI, you can convert multiple interfaces as members of a common broadcast domain. MPLS static over BVI feature allows you to specify a BVI interface as next hop while setting up a static LSP.

Only static MPLS tunnels can use BVI as a next hop. Also, a BVI next hop can be a static route, a directly connected route (IP address, not a subnet prefix), or a route resolved through BGP or IGP. The router will do an MPLS label lookup on incoming MPLS traffic, perform a label operation such as SWAP/PHP/POP, and forward the MPLS/IP traffic through the BVI next hop. The router can perform switching for Layer 2 traffic and routing for incoming Layer 3 MPLS traffic.

Restrictions

- If a BVI has multiple peers within a subnet, then the subnet prefix cannot be specified as the next hop IP address (though IP addresses within the subnet are BVI peers). You have to specify one of the peers (with a specific IP address) as the next hop.
- Back up paths over BVI (IPv4 or IPv6) are not supported.
- Fast Reroute (FRR) is not supported.
- Dynamic MPLS configuration is not supported. For example, label distribution using LDP is not supported.

Configuration Example

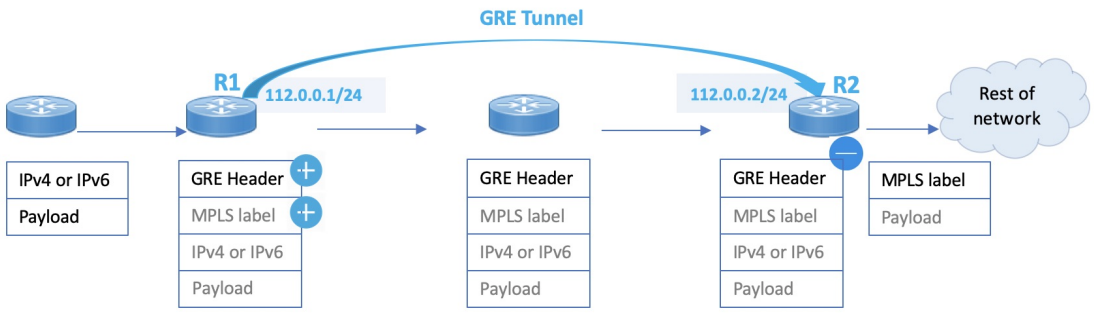
This example shows how to configure a BVI interface as next hop for a static LSP.

```
RP/0/0/CPU0:Router# configure terminal
RP/0/0/CPU0:Router(config)# mpls static
RP/0/0/CPU0:Router(config-mpls-static)# interface TenGig 0/0/0/0
RP/0/0/CPU0:Router(config-mpls-static)# lsp bvi
RP/0/0/CPU0:Router(config-mpls-static-lsp)# in-label 5001 allocate
RP/0/0/CPU0:Router(config-mpls-static-lsp)# forward
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 nexthop BVI1 192.168.2.1 out-label
pop
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# path 1 nexthop BVI1 192.168.2.1 out-label
4444
RP/0/0/CPU0:Router(config-mpls-static-lsp-fwd)# exit
```

MPLS Over Single-Pass GRE Tunnels

This feature supports MPLS static forwarding over a single-pass GRE tunnel at line rate. One use case is of a Provider router sending incoming customer traffic over the GRE tunnel, addressed to an anycast virtual IP address (VIP) destination shared by a set of load balancing servers.

Figure 10: MPLS Over a Single-Pass GRE Tunnel



In the image, you can see that the GRE tunnel begins at R1. R1 checks ACL configurations, adds an MPLS label to the incoming packet, and then adds a GRE header. Then it sends the traffic towards R2.

R2 uses the PBR process for GRE tunnel decapsulation, and based on the MPLS label, it forwards the traffic towards its destination.

Configuration Example

This example shows how to enable MPLS static forwarding over a single-pass GRE tunnel at line rate.

GRE Tunnel Configuration on R1

The single-pass GRE tunnel starts on R1.

```
R1# configure
R1(config)# interface tunnel-ip1
R1(config-if)# ipv4 address 112.0.0.1 255.255.255.0
R1(config-if)# tunnel mode gre ipv4 encap
R1(config-if)# tunnel source TenGigE0/0/0/2
R1(config-if)# tunnel destination 50.0.0.1
R1(config-if)# commit
```

GRE tunnel destination address is an anycast address. GRE encapsulation must be based on an ACL or a policy-map, or both. A destination can be an individual address or a /28 prefix.

MPLS Static Configuration on R1

```
R1# configure
R1(config)# router static
R1(config-static)# address-family ipv4 unicast
R1(config-static-afi)# 111.0.0.1/32 tunnel-ip1
R1(config-static-afi)# commit

R1(config)# mpls static
R1(config-mpls-static)# lsp test
R1(config-mpls-static-lsp)# in-label 10000 allocate per-prefix 111.0.0.1/32
R1(config-mpls-static-lsp)# forward
R1(config-mpls-static-lsp-fwd)# path 1 nexthop tunnel-ip1 out-label 12000
R1(config-mpls-static-lsp-fwd)# commit
```

GRE Tunnel Configuration on R2

The single-pass GRE tunnel terminates on R2.

```
R2 # configure
R2(config)# interface tunnel-ip1
R2(config-if)# ipv4 address 112.0.0.2 255.255.255.0
R2(config-if)# tunnel mode gre ipv4 decap
R2(config-if)# tunnel source TenGigE0/0/0/2
R2(config-if)# tunnel destination 10.0.0.1
R2(config-if)# commit
```

Verification

Tunnel-IP configuration on R1

```
R1# show running-config interface tunnel-ip 1

interface tunnel-ip1
    ipv4 address 112.0.0.1 255.255.255.0
    tunnel mode gre ipv4 encap
    tunnel source TenGigE0/0/0/2
    tunnel destination 50.0.0.1
!
```

MPLS Static Configuration on R1

```
R1# show running-config router static

router static
 address-family ipv4 unicast
  111.0.0.1/32 tunnel-ip1
 !
!
```

```
R1# show running-config mpls static

mpls static
  lsp test
     in-label 1000 allocate per-prefix 111.0.0.1/32
     forward
        path 1 nexthop tunnel-ip1 out-label 12000
        !
```

Tunnel-IP Configuration on R2

```
R2# show running-config int tunnel-ip 1

interface tunnel-ip1
 ipv4 address 112.0.0.2 255.255.255.0
 tunnel mode gre ipv4
 tunnel source TenGigE0/0/0/2
 tunnel destination 10.0.0.1
!
```

PBR Configuration for GRE Tunnel Decapsulation on R2

```
R2# show running-config class-map type traffic match-all

class-map type traffic match-all test_gre1
    match protocol gre
    match destination-address ipv4 50.0.0.1 255.255.255.255
    match source-address ipv4 10.0.0.1 255.255.255.255
end-class-map
!
policy-map type pbr P1-test
```

```
   class type traffic test_gre1
   decapsulate gre
!
   class type traffic class-default
!
end-policy-map
!

vrf-policy
vrf default address-family ipv4 policy type pbr input P1-test
!
```

Multicast Decapsulated Flow Over Static LSP GRE Tunnels

Table 13: Feature History Table

Feature Name	Release Information	Feature Description
Multicast Decapsulated Flow Over Static Label Switched Paths Over Single Pass GRE Tunnels	Release 7.10.1	Introduced in this release on: NCS 5500 fixed port routers This feature enables the device to forward the L2 frames payload from decapsulated static label-switched paths (LSPs) over single-pass GRE tunnels. You can now forward only the L2 frames from the decapsulated MPLS static LSPs that are carrying L2 frames. This feature introduces these changes: • The **l2-bypass** keyword is introduced in the **mpls static** command.

With this feature enabled, you can now configure your multicast MPLS static LSPs layer 2 (L2) packets over single-pass GRE tunnels, which are decapsulated to forward only the L2 frames as it is, through the egress interface.

Note This feature isn't supported on Cisco NCS 5700 Routers and Line cards.

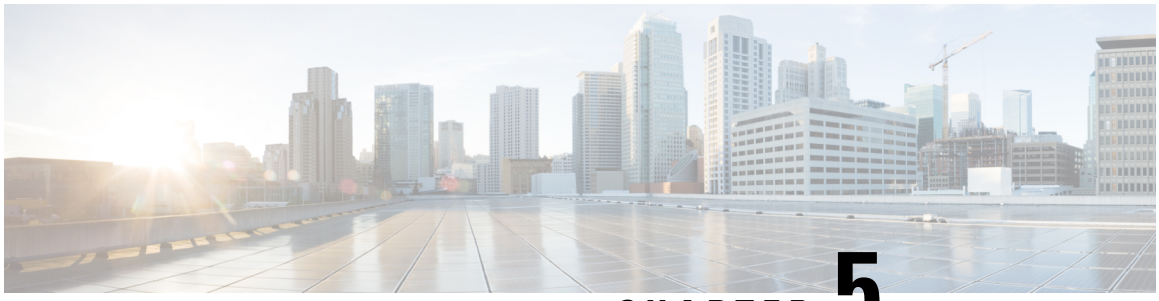
Configuration Example

Use the **l2-bypass** keyword under the **mpls static** command to configure this feature:

```
Router:# config
Router(config):# mpls static
Router(config-mpls-static):# lsp FOOBAR
Router(config-mpls-static):# in-label 13000 allocate
```

```
Router(config-mpls-static):# forward
Router(config-mpls-static-fwd):# path 1 nexthop Bundle-Ether333.2 10.0.0.2 out-label pop
Router(config-mpls-static-fwd):# l2-bypass
Router(config-mpls-static-fwd):# exit
```

In this example, incoming MPLS traffic with the label 13,000 will have the MPLS label decapsulated and the L2 frames will be sent as it's to the next hop 10.0.0.2 through the Bundle-Ether333.2 egress interface.



CHAPTER 5

Implementing RSVP for MPLS-TE

- Implementing RSVP for MPLS-TE , on page 65

Implementing RSVP for MPLS-TE

Resource Reservation Protocol (RSVP) is a signaling protocol that enables systems to request resource reservations from the network. RSVP processes protocol messages from other systems, processes resource requests from local clients, and generates protocol messages. As a result, resources are reserved for data flows on behalf of local and remote clients. RSVP creates, maintains, and deletes these resource reservations.

MPLS Traffic Engineering (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. MPLS-TE uses RSVP to signal LSPs.

Usage Guidelines and Limitations

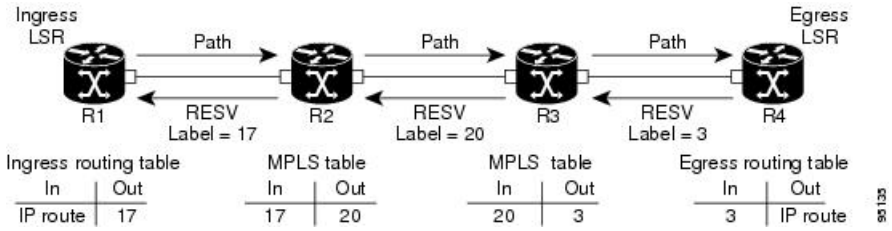
Consider these usage guidelines and limitations before implementing RSVP for MPLS-TE:

- To switch between an RSVP-TE or SR-TE configuration and an MPLS-TE configuration and vice versa, it is necessary to remove the configuration of the first process, say, RSVP-TE or SR-TE, in a separate commit operation and add the configuration of the next process, say, MPLS-TE, in another commit operation. This prevents any commit failures caused due to the delay in unconfiguring the first process before the next process configuration is applied.

Setting up MPLS LSP Using RSVP

The following figure shows how RSVP sets up a LSP from router R1 through router R4 that can be used for TE in an MPLS environment.

Figure 11: MPLS LSP Using RSVP



The LSP setup is initiated when the LSP head node sends path messages to the tail node. The Path messages reserve resources along the path to each node, and creates path states associated with the session on each node. When the tail node receives a path message, it sends a reservation (RESV) message with a label back to the previous node. 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.

When the reservation message arrives at the previous node, it causes the reserved resources to be locked and forwarding entries are programmed with the MPLS label sent from the tail-end node. A new MPLS label is allocated and sent to the next node upstream. When the reservation message reaches the head node, the label is programmed and the MPLS data starts to flow along the path.

Overview of RSVP for MPLS-TE Features

This section provides an overview of the various features of RSVP for MPLS-TE.

RSVP is automatically enabled on interfaces on which MPLS-TE is configured. For MPLS-TE LSPs with bandwidth, the RSVP bandwidth has to be configured on the interfaces. There is no need to configure RSVP, if all MPLS-TE LSPs have zero bandwidth.

RSVP Graceful restart ensures high availability and allows RSVP TE enabled routers to recover RSVP state information from neighbors after a failure in the network.

RSVP requires that the path and reservation state that are set up during LSP signaling must be refreshed by periodically sending refresh messages. Refresh messages are used to synchronize the state between RSVP neighbors and to recover from lost RSVP messages. RSVP refresh reduction feature includes support for reliable messages which are transmitted rapidly when the messages are lost. Summary refresh messages contain information to refresh multiple states and reduces the number of messages required to refresh states.

RSVP messages can be authenticated to ensure that only trusted neighbors can set up reservations.

For detailed information about RSVP for MPLS-TE features, see *RSVP for MPLS-TE Features- Details*.

Configuring RSVP for MPLS-TE

RSVP requires coordination among several routers, establishing exchange of RSVP messages to set up LSPs. To configure RSVP, you need to install the following two RPMs :

- ncs5500-mpls-2.0.0.0-r601.x86_64.rpm-6.0.1
- ncs5500-mpls-te-rsvp-2.0.0.0-r601.x86_64.rpm-6.0.1

Depending on the requirements, RSVP requires some basic configuration described in the following topics:

Configuring RSVP Message Authentication Globally

The RSVP authentication feature permits neighbors in an RSVP network to use a secure hash algorithm to authenticate all RSVP signaling messages digitally. The authentication is accomplished on a per-RSVP-hop basis using an RSVP integrity object in the RSVP message. The integrity object includes a key ID, a sequence number for messages, and keyed message digest.

You can globally configure the values of authentication parameters including the key-chain, time interval that RSVP maintains security associations with other trusted RSVP neighbors (life time) and maximum number of RSVP authenticated messages that can be received out of sequence (window size). These defaults are inherited for each neighbor or interface.

Configuration Example

In this example, authentication parameters are configured globally on a router. The authentication parameters including authentication key-chain, lifetime, and window size are configured. A valid key-chain should be configured before performing this task.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# key chain mpls-keys
RP/0/RP0/CPU0:router(config-mpls-keys)# exit
RP/0/RP0/CPU0:router(config)# rsvp authentication
RP/0/RP0/CPU0:router(config-rsvp-auth)# key-source key-chain mpls-keys
RP/0/RP0/CPU0:router(config-rsvp-auth)# life-time 2000
RP/0/RP0/CPU0:router(config-rsvp-auth)# window-size 33
```

Verification

Verify the configuration of authentication parameters using the following command.

```
RP/0/RP0/CPU0:router# show rsvp authentication detail

  RSVP Authentication Information:
    Source Address:         3.0.0.1
    Destination Address:    3.0.0.2
    Neighbour Address:      3.0.0.2
    Interface:              HundredGigabitEthernet 0/0/0/3
    Direction:              Send
    LifeTime:               2000 (sec)
    LifeTime left:          1305 (sec)
    KeyType:                Static Global KeyChain
    Key Source:             mpls-keys
    Key Status:             No error
    KeyID:                  1
    Digest:                 HMAC MD5 (16)
    window-size:            33
Challenge:              Not supported
    TX Sequence:            5023969459702858020 (0x45b8b99b00000124)
    Messages successfully authenticated:    245
    Messages failed authentication:         0
```

Related Topics

- Configuring RSVP Authentication for an Interface, on page 68
- Configuring RSVP Authentication on a Neighbor, on page 68
- #unique_67

Configuring RSVP Authentication for an Interface

You can individually configure the values of RSVP authentication parameters including key-chain, life time, and window size on an interface. Interface specific authentication parameters are used to secure specific interfaces between two RSVP neighbors.

Configuration Example

This example configures authentication key-chain, life time for the security association, and window size on an interface. A valid key-chain should be already configured to use it as part of this task.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# authentication
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# key-source key-chain mpls-keys
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# life-time 2000
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# window-size 33
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the configuration of authentication parameters using the following command.

```
RP/0/RP0/CPU0:router# show rsvp authentication detail

  RSVP Authentication Information:
    Source Address:          3.0.0.1
    Destination Address:     3.0.0.2
    Neighbour Address:       3.0.0.2
    Interface:               HundredGigabitEthernet 0/0/0/3
    Direction:               Send
    LifeTime:                2000 (sec)
    LifeTime left:           1305 (sec)
    KeyType:                 Static Global KeyChain
    Key Source:              mpls-keys
    Key Status:              No error
    KeyID:                   1
    Digest:                  HMAC MD5 (16)
    window-size:             33
    Challenge:               Not supported
    TX Sequence:             5023969459702858020 (0x45b8b99b00000124)
    Messages successfully authenticated:    245
    Messages failed authentication:         0
```

Related Topics

- Configuring RSVP Message Authentication Globally, on page 67
- Configuring RSVP Authentication on a Neighbor, on page 68
- #unique_67

Configuring RSVP Authentication on a Neighbor

You can individually configure the values of RSVP authentication parameters including key-chain, life time, and window size on a neighbor.

Configuration Example

This example configures the authentication key-chain, life time for the security association, and window size on a RSVP neighbor. A valid key-chain should be already configured to use it as part of this task.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp neighbor 10.0.0.1 authentication
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# key-source key-chain mpls-keys
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# life-time 2000
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# window-size 33
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the configuration of authentication parameters using the following command.

```
RP/0/RP0/CPU0:router# show rsvp authentication detail

  RSVP Authentication Information:
    Neighbour Address:      10.0.0.1
    Interface:              HundredGigabitEthernet 0/0/0/3
    Direction:              Send
    LifeTime:               2000 (sec)
    LifeTime left:          1205 (sec)
    KeyType:                Static Global KeyChain
    Key Source:             mpls-keys
    Key Status:             No error
    KeyID:                  1
    Digest:                 HMAC MD5 (16)
    window-size:            33
    Challenge:              Not supported
```

Related Topics

- Configuring RSVP Message Authentication Globally, on page 67
- Configuring RSVP Authentication for an Interface, on page 68
- #unique_67

MPLS-TE PCE authentication

The PCEP session authentication is a security mechanism that

- establishes a secure and trusted communication channel between a PCC and a PCE,
- ensures that only authorized devices can participate in PCEP sessions, and
- protects against unauthorized access and data tampering.

Methods of MPLS-TE PCE authentication

You can perform MPLS-TE PCE authentication using TCP Message Digest 5 (MD5) authentication. This method uses a clear text or encrypted password for authentication. The Cisco IOS XR software rejects segments that lack a Message Authentication Code (MAC) matching the configured password.

Before Release 26.1.1, you can perform MPLS-TE PCE authentication using TCP Message Digest 5 (MD5) authentication. This method uses a clear text or encrypted password for authentication. The Cisco IOS XR software rejects segments that lack a Message Authentication Code (MAC) matching the configured password.

From Release 26.1.1, you can perform MPLS-TE PCE authentication using TCP Authentication Option (TCP-AO).

TCP-AO uses Message Authentication Codes (MACs), which provides the following benefits:

- It protects against replay attacks for long-lived TCP connections.
- It provides more detailed security association with TCP connections than TCP MD5.
- It supports a larger set of MAC algorithms with minimal system and operational changes.

TCP-AO is compatible with Master Key Tuple (MKT) configuration, protecting connections by deriving traffic keys from the MKT and coordinating changes between endpoints. The Cisco IOS XR software rejects segments that lack a MAC matching the configured key chain.

Note TCP-AO and TCP MD5 are never permitted to be used simultaneously. TCP-AO supports IPv6, and is fully compatible with the proposed requirements for the replacement of TCP MD5.

Configure MPLS-TE PCE authentication

Procedure

Step 1 Before Release 26.1.1, you can configure MPLS-TE PCE authentication using TCP MD5 authentication. Specify if the password is encrypted or clear text.

Example:

```
Router# configure terminal
Router(config)# mpls traffic-eng
Router(config-mpls-te)# pce password [{clear | encrypted} <password>]
```

Step 2 From Release 26.1.1, you can configure MPLS-TE PCE authentication using TCP Authentication Option (TCP-AO). Use the **include-tcp-options** keyword to include other TCP options in the header for MAC calculation.

Example:

```
Router# configure terminal
Router(config)# mpls traffic-eng
Router(config-mpls-te)# pce tcp-ao <password> include-tcp-options
```

Step 3 Run the **show mpls traffic-eng pce peer** command to verify the PCE up status.

Example:

```
Router# show mpls traffic-eng pce peer

Address Precedence State Learned From
--------------- ------------ ------------ --------------------
192.168.1.4 255 Up Static config
```

Step 4 Run the **show tcp detail** command to verify the TCP session details.

Example:

```
Router# show tcp detail pcb all

==============================================================
Connection state is ESTAB, I/O status: 0, socket status: 0
Established at Thu Feb 26 05:36:17 2026

PCB 0x00007fc85c0049d0, SO 0x7fc85c0045b0, TCPCB 0x7fc85c004ce0, vrfid 0x60000000,
Pak Prio: Medium, TOS: 192, TTL: 255, Hash index: 1854
```

```
Local host: 192.168.0.1, Local port: 52364 (Local App PID: 8630)
Foreign host: 192.168.1.4, Foreign port: 4189
(Local App PID/instance/SPL_APP_ID: 8630/0/0)

Current send queue size in bytes: 0 (max 512000)
Current receive queue size in bytes: 0 (max 512000) mis-ordered: 0 bytes
Current receive queue size in packets: 0 (max 60)


Num of peers with authentication info: 1
Authentication peer details:
Peer: 192.168.1.4/32, OBJ_ID: 0x40002fc0
Port: 4189, vrf_id: 0x60000000, type: AO, debug_on:0
Keychain_name: test11, options: 0x00000001, linked peer: 0x7fc864004770
Send_SNE: 0, Receive_SNE: 0, Send_SNE_flag: 0
Recv_SNE_flag: 0, Prev_send_seq: 3997493771, Prev_receive_seq: 401915492
ISS: 3997493682, IRS: 401915403
Current key: 1
Traffic keys: send_non_SYN: ffd64ea3, recv_non_SYN: 00000000
RNext key: 1
Traffic keys: send_non_SYN: 00000000, recv_non_SYN: 00000000
Last 1 keys used:
key: 1, time: Feb 26 05:36:17.344.766, reason: No current key set
```

The above sample displays only a part of the actual output; the actual output displays more details.

Configuring Graceful Restart

RSVP graceful restart provides a mechanism to ensure high availability (HA), which allows detection and recovery from failure conditions for systems running cisco IOS XR software and ensures non-stop forwarding services. RSVP graceful restart is based on RSVP hello messages and allows RSVP TE enabled routers to recover RSVP state information from neighbors after a failure in the network. RSVP uses a Restart Cap object (RSVP RESTART) in hello messages in which restart and recovery times are specified to advertise the restart capability of a node. The neighboring node helps a restarting node by sending a Recover Label object to recover the forwarding state of the restarting node.

You can configure standard graceful restart which is based on node-id address based hello messages and also interface-based graceful restart which is interface-address based hello messages.

Configuration Example

In this example, RSVP-TE is already enabled on the router nodes on a network and graceful restart needs to be enabled on the router nodes for failure recovery. Graceful restart is configured globally to enabled node-id address based hello messages and also on a router interface to support interface-address based hello messages.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# signalling graceful-restart
RP/0/RP0/CPU0:router(config-rsvp)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling graceful-restart
interface-based
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Use the following commands to verify that graceful restart is enabled.

```
RP/0/RP0/CPU0:router# show rsvp graceful-restart
 Graceful restart: enabled Number of global neighbors: 1
 Local MPLS router id: 192.168.55.55
 Restart time: 60 seconds  Recovery time: 120 seconds
 Recovery timer: Not running
 Hello interval: 5000 milliseconds Maximum Hello miss-count: 4

RP/0/RP0/CPU0:router# show rsvp graceful-restart neighbors detail

 Neighbor: 192.168.77.77 Source: 192.168.55.55 (MPLS)
   Hello instance for application MPLS
     Hello State: UP        (for 00:20:52)
     Number of times communications with neighbor lost: 0
     Reason: N/A
Recovery State: DONE
   Number of Interface neighbors: 1
     address: 192.168.55.0
   Restart time: 120 seconds  Recovery time: 120 seconds
   Restart timer: Not running
   Recovery timer: Not running
   Hello interval: 5000 milliseconds  Maximum allowed missed Hello messages: 4
```

Related Topics

- #unique_67

Configuring Refresh Reduction

RSVP Refresh Reduction improves the reliability of Resource Reservation Protocol (RSVP) signaling to enhance network performance and message delivery and it is enabled by default. Refresh reduction is used with a neighbor only if the neighbor supports it. You can also disable refresh reduction on an interface if you want.

This feature ensures reliable delivery of RSVP messages when network traffic is disrupted. To ensure that its message is delivered to its neighbor, RSVP requests the neighbor to send an acknowledgment message by a given time duration. If it doesn't receive the acknowledgment, it resends the message and doubles its current wait time. After 5 attempts, RSVP stops retransmitting the message to the neighbor.

Configuration Example

The example shows how to configure the various parameters available for the refresh reduction feature.

The following parameters are configured to change their default values:

- refresh interval
- number of refresh messages a node can miss
- retransmit time
- acknowledgment hold time
- acknowledgment message size
- refresh message summary size

```
Router# configure
Router(config)# rsvp
Router(config-rsvp)# interface HundredGigabitEthernet 0/0/0/3
Router(config-rsvp-if)# signalling refresh interval 40
Router(config-rsvp-if)# signalling refresh missed 6
Router(config-rsvp-if)# signalling refresh reduction reliable retransmit-time 2000
Router(config-rsvp-if)# signalling refresh reduction reliable ack-hold-time 1000
Router(config-rsvp-if)# signalling refresh reduction reliable ack-max-size 1000
Router(config-rsvp-if)# signalling refresh reduction summary  max-size 1500
  Router(config)# commit
```

Configuring ACL Based Prefix Filtering

You can configure extended access lists (ACLs) to forward, drop, or perform normal processing on RSVP router-alert (RA) packets. For each incoming RSVP RA packet, RSVP inspects the IP header and attempts to match the source or destination IP addresses with a prefix configured in an extended ACL. If there is no explicit permit or explicit deny, the ACL infrastructure returns an implicit deny by default. By default, RSVP processes the packet if the ACL match yields an implicit (default) deny.

Configuration Example

This example configures ACL based prefix filtering on RSVP RA packets. When RSVP receives a RA packet from source address 10.0.0.1 it is forwarded and packets destined to the IP address 172.16.0.1 are dropped.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 access-list rsvpac1
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 host 10.0.0.1 any
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny ipv4 any host 172.16.0.1
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# signalling prefix-filtering access-list rsvp-acl
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the configuration of ACL based prefix filtering

```
RP/0/RP0/CPU0:router# show rsvp counters prefix-filtering access-list rsvp-ac1

  ACL:rsvp-ac1            Forward          Local          Drop           Total
  Path                      0              0            0              0
  PathTear                  0              0            0              0
  ResvConfirm               0              0            0              0
  Total                     0              0            0
```

Related Topics

- Configuring RSVP Packet Dropping, on page 73

Configuring RSVP Packet Dropping

You can configure extended access lists (ACLs) to forward, drop, or perform normal processing on RSVP router-alert (RA) packets. By default, RSVP processes the RA packets even if the ACL match yields an implicit deny. You can configure RSVP to drop RA packets when the ACL matches results in an implicit deny.

Configuration Example

This example configures ACL based prefix filtering on RSVP RA packets. When RSVP receives a RA packet from source address 10.0.0.1 it is forwarded and packets destined to the IP address 172.16.0.1 are dropped. RA packets are dropped if the ACL matches results in an implicit deny.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 access-list rsvpac1
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 host 10.0.0.1 any
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny ipv4 any host 172.16.0.1
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# signalling prefix-filtering default-deny-action drop
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the configuration of RSVP packet drop using the following command.

```
RP/0/RP0/CPU0:router# show rsvp counters prefix-filtering access-list rsvpac1

  ACL: rsvpac1            Forward          Local          Drop           Total
  Path                      4              1              0              5
  PathTear                  0              0              0              0
  ResvConfirm               0              0              0              0
  Total                     4              1              0              5
```

Related Documents

- Configuring ACL Based Prefix Filtering, on page 73

Enabling RSVP Traps

By implementing the RSVP MIB, you can use SNMP to access objects belonging to RSVP. You can also specify two traps (NewFlow and LostFlow) which are triggered when a new flow is created or deleted. RSVP MIBs are automatically enabled when you turn on RSVP, but you need to enable RSVP traps.

Configuration Example

This example shows how to enable RSVP MIB traps when a flow is deleted or created and also how to enable both the traps.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp lost-flow
RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp new-flow
RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp all
RP/0/RP0/CPU0:router(config)# commit
```

RSVP for MPLS-TE Features- Details

RSVP Graceful Restart Operation

RSVP graceful restart is based on RSVP hello messages. Hello messages are exchanged between the router and its neighbor nodes. Each neighbor node can autonomously issue a hello message containing a hello request object. A receiver that supports the hello extension replies with a hello message containing a hello acknowledgment (ACK) object. If the sending node supports state recovery, a Restart Cap object that indicates a node's restart capability is also carried in the hello messages. In the Restart Cap object, the restart time and the recovery time is specified. The restart time is the time after a loss in Hello messages within which RSVP

hello session can be re-established. The recovery time is the time that the sender waits for the recipient to re-synchronize states after the re-establishment of hello messages.

For graceful restart, the hello messages are sent with an IP Time to Live (TTL) of 64. This is because the destination of the hello messages can be multiple hops away. If graceful restart is enabled, hello messages (containing the restart cap object) are send to an RSVP neighbor when RSVP states are shared with that neighbor. If restart cap objects are sent to an RSVP neighbor and the neighbor replies with hello messages containing the restart cap object, the neighbor is considered to be graceful restart capable. If the neighbor does not reply with hello messages or replies with hello messages that do not contain the restart cap object, RSVP backs off sending hellos to that neighbor. If a hello Request message is received from an unknown neighbor, no hello ACK is sent back.

RSVP Authentication

Network administrators need the ability to establish a security domain to control the set of systems that initiates RSVP requests. The RSVP authentication feature permits neighbors in an RSVP network to use a secure hash to sign all RSVP signaling messages digitally, thus allowing the receiver of an RSVP message to verify the sender of the message without relying solely on the sender's IP address.

The signature is accomplished on a per-RSVP-hop basis with an RSVP integrity object in the RSVP message as defined in RFC 2747. The integrity object includes a key ID, a sequence number for messages, and keyed message digest. This method provides protection against forgery or message modification. However, the receiver must know the security key used by the sender to validate the digital signature in the received RSVP message. Network administrators manually configure a common key for each RSVP neighbor on the shared network. The sending and receiving systems maintain a security association for each authentication key that they share. For detailed information about different security association parameters, see the **Security Association Parameters** table.

You can configure global defaults for all authentication parameters including key, window size, and lifetime. These defaults are inherited when you configure authentication for each neighbor or interface. However, you can also configure these parameters individually on a neighbor or interface basis, in which case the global values (configured or default) are no longer inherited.

Interface and neighbor interface modes unless explicitly configured, inherit the parameters from global configuration mode as follows:

- Window-size is set to 1.
- Lifetime is set to 1800.
- key-source key-chain command is set to none or disabled.

The following situations explain how to choose between global, interface, or neighbor configuration modes:

- Global configuration mode is optimal when a router belongs to a single security domain (for example, part of a set of provider core routers). A single common key set is expected to be used to authenticate all RSVP messages.
- Interface, or neighbor configuration mode, is optimal when a router belongs to more than one security domain. For example, a provider router is adjacent to the provider edge (PE), or a PE is adjacent to an edge device. Different keys can be used but not shared.

A security association (SA) is a collection of information that is required to maintain secure communications with a peer. The following table lists the main parameters that defines a security association

Table 14: Security Association Parameters

Security Association Parameters	Description
src	IP address of the sender.
dst	IP address of the final destination.
interface	Interface of the security association.
direction	Send or receive type of the security association.
Lifetime	Expiration timer value that is used to collect unused security association data.
Sequence Number	Last sequence number that was either sent or accepted (dependent of the direction type).
key-source	Source of keys for the configurable parameter.
keyID	Key number (returned form the key-source) that was last used.
Window Size	Specifies the maximum number of authenticated messages that can be received out of order.
Window	Specifies the last *window size* value sequence number that is received or accepted.

Configuring Midpoint LSP limit (LSP out-of-resource [OOR])

You can limit the RSVP-TE states on an LSR. To achieve this, specify an upper limit on the number of midpoint LSPs, and a limit on the unprotected LSPs on the LSR.

The actions that are triggered when the resource exhaustion happens allows you to fine tune the network to avoid the router in question.

You can specify Yellow and Red thresholds for midpoint LSP limits (overall and unprotected). Each limit has a value for the Yellow threshold and the Red threshold.

When the thresholds are crossed, the actions configured under the LSP-OOR sub-mode take effect. These actions are applied on *all* the interfaces of the LSR. The same timed transition to Green state will also occur when the LSP numbers drop.

The limits do not apply to ingress and egress LSPs as they are driven by explicit configuration. In other words, the configuration determines how many egress or ingress LSPs a given node has. For midpoints, it is different as the number is a function of the topology, the links metrics and links bandwidth.

State Transition Triggers - The LSP-OOR state transition is triggered by checking both the total transit LSP count and the unprotected count. If either count crosses the threshold, then the state transition is triggered. If both counts cross the limit, then the more critical state is chosen.

Low vs High Priority LSPs - when the LSP OOR is in yellow or red state, new high priority LSPs will not pre-empt low priority LSP. Pre-emption can still occur but only for bandwidth reasons. In other words, if the router is in Red state where one of the actions is to reject any new LSP, then new high-priority LSPs will be

rejected even if there is already an established low-priority LSP: the low-priority LSP will not be removed to make room for the high-priority one.

Configuration Change - Setting the configured limit to a value that is smaller than the current number of LSPs will not cause existing LSPs to be deleted or pre-empted.

Setting the configured limit to a value that is larger than the current number of LSPs takes the node out of LSP-OOR state.

Path Error - When an LSP cannot be admitted due to LSP-OOR, the path-error is sent to the LER.

Interaction between LSP-OOR and HW-OOR - When both HW-OOR and LSP-OOR are active, the more critical action is done based on the interface in question. For example, if the LSP-OOR is Red and HW-OOR is Yellow for interface A, then TE will apply the Red state actions for all interfaces including interface A. Conversely, if the LSP-OOR is Yellow for the overall LSPs and HW-OOR is Red for interface A, then TE will apply the Yellow state actions for all interfaces except interface A. Instead, TE will apply the Red state actions for interface A.

Configuration Example

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# lsp-oor
Router(config-te-lsp-oor)# yellow
Router(config-te-lsp-oor-state)# transit-all threshold 45000
Router(config-te-lsp-oor-state)# transit-unprotected threshold 15000
Router(config-te-lsp-oor-state)# exit

Router(config-te-lsp-oor)# red
Router(config-te-lsp-oor-state)# transit-all threshold 50000
Router(config-te-lsp-oor-state)# transit-unprotected threshold 20000
Router(config-te-lsp-oor-state)# commit
```

The defaults of the above thresholds are infinite.

When a threshold is crossed, these below actions are applied to all the TE interfaces on the box. When the “accept-reopt” configuration is added it will work for both HW-OOR and LSP-OOR.

```
Router(config-mpls-te)# hw-oor
Router(config-te-hwoor)# green
Router(config-te-hwoor-state)# action flood available-bw 80
Router(config-te-hwoor-state)# recovery-duration 1
Router(config-te-hwoor-state)# action admit lsp-min-bw 1000
Router(config-te-hwoor-state)# action flood te-metric penalty 100
Router(config-te-hwoor-state)# action node-protection disable
Router(config-te-hwoor-state)# exi
Router(config-te-hwoor)# yellow
Router(config-te-hwoor-state)# action flood available-bw 50
Router(config-te-hwoor-state)# action admit lsp-min-bw 5000
Router(config-te-hwoor-state)# action flood te-metric penalty 200
Router(config-te-hwoor-state)# action node-protection disable
Router(config-te-hwoor-state)# exi
Router(config-te-hwoor)# red
Router(config-te-hwoor-state)# action flood available-bw 0
Router(config-te-hwoor-state)# action admit lsp-min-bw 10000
Router(config-te-hwoor-state)# action flood te-metric penalty 1000
Router(config-te-hwoor-state)# action node-protection disable
Router(config-te-hwoor-state)# exi
Router(config-te-hwoor)# commit
```

Verification

```
Router# show mpls traffic-eng lsp-oor summary

Total Transit LSP:

Yellow Threshold: 45000
  Red Threshold:    50000
  Current Count:    23456

Unprotected Transit LSP:

Yellow Threshold: 15000
  Red Threshold:    20000
  Current Count:    14456

LSP OOR Status: Green; Changed last at: Thu Oct 27 16:41:22 2016

Green state recovery actions in effect for: 115 seconds

LSP OOR Green State Parameters:
  Available Bandwidth percentage: 90%
  TE Metric Penalty: 0
  Minimum LSP Size: 0 kbps
  Transition duration: 2 minutes
  Statistics:
    Transitions 0; LSPs accepted 0, rejected 0;

LSP OOR Yellow State Parameters:
  Available Bandwidth percentage: 50%
  TE Metric Penalty: 10
  Minimum LSP Size: 500000 kbps
  HW OOR Statistics:
    Transitions 0; LSPs accepted 0, rejected 0;

LSP OOR Red State Parameters:
  Available Bandwidth percentage: 5%
  TE Metric Penalty: 50
  Minimum LSP Size: 1000000 kbps
  HW OOR Statistics:
    Transitions 0; LSPs accepted 0, rejected 0;
```

MPLS-TE LSP OOR

The MPLS-TE LSP OOR function adds capability for the RSVP-TE control plane to track the LSP scale of transit routers, so that it can take a specific set of (pre-configured) actions when threshold limits are crossed, and inform other routers in the network. MPLS-TE keeps track of the number of transit LSPs set up through the router. The limits do not apply to ingress and egress LSP routers since they are driven by explicit configuration. In other words, the configuration determines how many egress or ingress LSPs a router has. For midpoint routers, the number is a function of the topology, the links metrics, and links' bandwidth.

State Transition Triggers - The LSP OOR state transition is triggered by checking the total transit LSP count and the unprotected count. If either count crosses the threshold, the state transition is triggered. If both counts cross the limit, the more critical state is chosen. Each limit will have a value for the *Yellow* threshold and a value for the *Red* threshold. When these thresholds are crossed, the configured MPLS-TE LSP OOR actions take effect. Similarly, the transition to *Green* state occurs when the LSP numbers drop.

LSP OOR State Dampening - The reason for LSP OOR State Dampening is that the number of accepted LSPs would be at the threshold and once an LSP is deleted, the state goes back from Red to Yellow, and a new LSP is setup and the state goes back to Red.

The solution is to introduce dampening when there is a state transition from Red to Yellow or from Yellow to Green. Whenever the transit number of LSPs crosses down a threshold, a timer is started for 10 seconds. After the timer expires, the new state is computed and moved to it. The timer is stopped if the transit number threshold is crossed (up) again. The transition from a state to a more severe state is not dampened.

Low and High Priority LSPs - When the LSP OOR is in yellow or red state, new high priority LSPs will not preempt low priority LSPs. Preemption can still occur but only for bandwidth reasons. In other words, if the router is in Red state where one of the actions is to reject any new LSP, the new high-priority LSPs are rejected even if there is an established low-priority LSP. The low-priority LSP is not removed to make room for the high-priority one.

Configuration Limit - Setting the configured limit to a value that is smaller than the current number of LSPs will trigger state transition but will not cause existing LSPs to be deleted or preempted. Setting the configured limit to a value that is larger than the current number of LSPs takes the node out of LSP OOR state. When an LSP cannot be admitted due to LSP OOR, the LSRs send Path Error messages to the LERs.

Event Logging - This is generated when the system transitions across OOR states, such as a resource change into an *yellow* or *red* state. Reporting level for *Red* is critical (1), and for yellow is warning (4). The following example shows that the count has crossed the threshold of 5000.

```
RP/0/RP1/CPU0:May 15 17:05:48 PDT: te_control[1034]: %ROUTING-MPLS_TE-4-LSP_OOR :

Transit LSP resources changed to Yellow.
Total transit: configured threshold 5000; actual count 5001;
Unprotected transit: configured threshold 4294967295; actual count 0
```

When the resource comes out of OOR, it will report as *green*.

Configuration Example

```
mpls traffic-eng
 lsp-oor
  green
   action accept reopt-lsp
   action flood available-bw 20
   recovery-duration
   action admit lsp-min-bw   X   -- > (in kbps, a lower limit than yellow and red state)

  yellow
   transit-all threshold 75000
   action accept reopt-lsp
   action flood available-bw 0
   action admit lsp-min-bw  Y

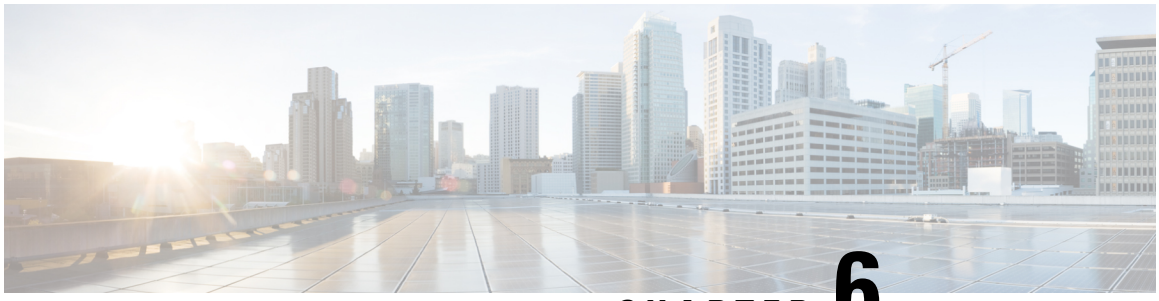
  red
   transit-all threshold 90000
   action flood available-bw 0
   action admit lsp-min-bw Z
```

The LSP OOR threshold values are set to yellow as 75000 and red as 90000. When these thresholds are crossed, corresponding actions are applied to all the TE interfaces.

Note The default values of the above thresholds are infinite.

When the LSP OOR *yellow* state is reached, the **accept reopt-lsp** action, **flood available-bw 0** action and **admit lsp-min-bw** actions are activated. This allows headend routers to reoptimize existing LSPs through, but doesn't allow new LSPs to get established. Also, MPLS-TE advertises zero bandwidth out of all interfaces, making this transit router less preferable for new LSPs. To handle a sudden burst of new LSPs that get signaled,

the **action admit lsp-min-bw** function ensures only a small number of high bandwidth LSPs get provisioned through the affected router. When the red threshold state is crossed, the **flood available-bw 0** and **admit lsp-min-bw** actions prevent any additional or reoptimized transit LSPs from getting set up through the affected router.



CHAPTER 6

Implementing MPLS Traffic Engineering

• Implementing MPLS Traffic Engineering, on page 81

Implementing MPLS Traffic Engineering

Traditional IP routing emphasizes on forwarding traffic to the destination as fast as possible. As a result, the routing protocols find out the least-cost route according to its metric to each destination in the network and every router forwards the packet based on the destination IP address and packets are forwarded hop-by-hop. Thus, traditional IP routing does not consider the available bandwidth of the link. This can cause some links to be over-utilized compared to others and bandwidth is not efficiently utilized. Traffic Engineering (TE) is used when the problems result from inefficient mapping of traffic streams onto the network resources. Traffic engineering allows you to control the path that data packets follow and moves traffic flows from congested links to non-congested links that would not be possible by the automatically computed destination-based shortest path.

Multiprotocol Label Switching (MPLS) with its label switching capabilities, eliminates the need for an IP route look-up and creates a virtual circuit (VC) switching function, allowing enterprises the same performance on their IP-based network services as with those delivered over traditional networks such as Frame Relay or Asynchronous Transfer Mode (ATM). MPLS traffic engineering (MPLS-TE) relies on the MPLS backbone to replicate and expand upon the TE capabilities of Layer 2 ATM and Frame Relay networks.

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. A router through which the tunnel passes is called the mid-point of the tunnel.

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).

Note Combination of unlabelled paths protected by labelled paths is not supported.

Overview of MPLS-TE Features

In MPLS traffic engineering, IGP extensions flood the TE information across the network. Once the IGP distributes the link attributes and bandwidth information, the headend router calculates the best path from head to tail for the MPLS-TE tunnel. This path can also be configured explicitly. Once the path is calculated, RSVP-TE is used to set up the TE LSP (Labeled Switch Path).

To forward the traffic, you can configure autoroute, forward adjacency, or static routing. The autoroute feature announces the routes assigned by the tailend router and its downstream routes to the routing table of the headend router and the tunnel is considered as a directly connected link to the tunnel.

If forward adjacency is enabled, MPLS-TE tunnel is advertised as a link in an IGP network with the link's cost associated with it. Routers outside of the TE domain can see the TE tunnel and use it to compute the shortest path for routing traffic throughout the network.

MPLS-TE provides protection mechanism known as fast reroute to minimize packet loss during a failure. For fast reroute, you need to create back up tunnels. The autotunnel backup feature enables a router to dynamically build backup tunnels when they are needed instead of pre-configuring each backup tunnel and then assign the backup tunnel to the protected interfaces.

DiffServ Aware Traffic Engineering (DS-TE) enables you to configure multiple bandwidth constraints on an MPLS-enabled interface to support various classes of service (CoS). These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint.

The MPLS traffic engineering autotunnel mesh feature allows you to set up full mesh of TE tunnels automatically with a minimal set of MPLS traffic engineering configurations. The MPLS-TE auto bandwidth feature allows you to automatically adjusts bandwidth based on traffic patterns without traffic disruption.

The MPLS-TE interarea tunneling feature allows you to establish TE tunnels spanning multiple Interior Gateway Protocol (IGP) areas and levels, thus eliminating the requirement that headend and tailend routers should reside in a single area.

For detailed information about MPLS-TE features, see MPLS-TE Features - Details, on page 132.

Note MPLS-TE Nonstop Routing (NSR) is enabled by default without any user configuration and cannot be disabled.

MPLS-TE NSR means the application is in hot-standby mode and standby MPLS-TE instance is ready to take over from the active instance quickly on RP failover.

Note that the MPLS-TE does not do routing. If there is standby card available then the MPLS-TE instance is in a hot-standby position.

The following output shows the status of MPLS-TE NSR:

```
Router#show mpls traffic-eng nsr status

TE Process Role        : V1 Active
Current Status       : Ready
    Ready since        : Tue Nov 01 10:42:34 UTC 2022 (1w3d ago)
   IDT started         : Tue Nov 01 03:28:48 UTC 2022 (1w3d ago)
    IDT ended          : Tue Nov 01 03:28:48 UTC 2022 (1w3d ago)
Previous Status        : Not ready
    Not ready reason   : Collaborator disconnected
    Not ready since    : Tue Nov 01 10:42:34 UTC 2022 (1w3d ago)
```

During any issues with the MPLS-TE, the NSR on the router gets affected which is displayed in the show redundancy output as follows:

```
Router#show mpls traffic-eng nsr status details
.
.
.

Current active rmf state: 4 (I_READY)
All standby not-ready bits clear - standby should be ready

Current active rmf state for NSR: Not ready
<jid> <node> <name> Reason for standby not NSR-ready
1082 0/RP0/CPU0 te_control TE NSR session not synchronized
Not ready set Wed Nov 19 17:28:14 2022: 5 hours, 23 minutes ago
1082 0/RP1/CPU0 te_control Standby not connected
Not ready set Wed Nov 19 17:29:11 2022: 5 hours, 22 minutes ago
```

Note If you have disabled MPLS-TE NSR and configured only MPLS-TE nonstop forwarding (NSF), performing an RP failover results in a verification error causing unprotected LSPs to flap and triggering FRR on protected LSPs. To avoid the verification error, you must block the path verification using the **mpls traffic-eng topology block-verify-after-rpfo** command.

How MPLS-TE Works

MPLS-TE automatically establishes and maintains label switched paths (LSPs) across the backbone by using RSVP. The path that an LSP uses is determined by the LSP resource requirements and network resources, such as bandwidth. Available resources are flooded by extensions to a link state based Interior Gateway Protocol (IGP). MPLS-TE tunnels are calculated at the LSP headend router, based on a fit between the required and available resources (constraint-based routing). The IGP automatically routes the traffic to these LSPs. Typically, a packet crossing the MPLS-TE backbone travels on a single LSP that connects the ingress point to the egress point.

The following sections describe the components of MPLS-TE:

Tunnel Interfaces

From a Layer 2 standpoint, an MPLS tunnel interface represents the headend of an LSP. It is configured with a set of resource requirements, such as bandwidth and media requirements, and priority. From a Layer 3 standpoint, an LSP tunnel interface is the headend of a unidirectional virtual link to the tunnel destination.

MPLS-TE Path Calculation Module

This calculation module operates at the LSP headend. The module determines a path to use for an LSP. The path calculation uses a link-state database containing flooded topology and resource information.

RSVP with TE Extensions

RSVP operates at each LSP hop and is used to signal and maintain LSPs based on the calculated path.

MPLS-TE Link Management Module

This module operates at each LSP hop, performs link call admission on the RSVP signaling messages, and keep track on topology and resource information to be flooded.

Link-state IGP

Either Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF) can be used as IGPs. These IGPs are used to globally flood topology and resource information from the link management module.

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.

RSVP-TE Bandwidth Accounting

The total interface bandwidth utilization in the data plane, excluding RSVP-TE bandwidth, is called dark bandwidth. A network may have dark bandwidth due to IP, LDP, or segment routing traffic flowing in the network. Dark bandwidth effectively reduces the link bandwidth available to RSVP-TE LSPs.

When segment routing is enabled on the network, you should be aware of the segment routing traffic over the links so that RSVP-TE bandwidth reservations can avoid overbooking the links in the network. The RSVP-TE bandwidth accounting feature allows you to perform proper accounting of the traffic.

You can enable RSVP-TE bandwidth accounting, and start dark bandwidth advertisement based on accounting samples, for all MPLS-TE enabled links using the **bandwidth-accounting** command in MPLS-TE configuration mode.

Computing the Effective Maximum Reservable Bandwidth

The statistics collector process (statsD) is responsible for returning statistics counters for each feature. For each traffic engineering (TE)-enabled interface, the TE process collects new RSVP-TE bandwidth rate statistics (samples) from the statsD process, within a specified sampling interval. These samples are collected over a period of time called an application interval.

After each application interval, the average value of the collected rate samples is used to compute the dark bandwidth rate and the effective maximum reservable bandwidth (Max-Reservable-BW) rate.

The dark bandwidth rate is calculated as total rate - RSVP rate:

- The total rate includes IPv4 , IPv6, and MPLS rate (MPLS includes RSVP/LDP/BGP label packets)
- The RSVP rate includes IPv4/IPv6 encapsulation over tunnel, and MPLS label packet counters for RSVP-TE tunnels

The following example shows how the effective maximum reservable bandwidth (Effective-Max-Reservable-BW) is computed (assuming a link capacity of 10Gbps and a configured RSVP bandwidth of 90%):

- Link capacity = 10Gbps
- Max-reservable-bw = RSVP percentage of link capacity = 9Gbps
- Total rate (from statsD) = 5Gpbs
- RSVP rate (from statsD) = 3Gpbs
- Dark-bw = Total rate - RSVP rate = 2Gpbs
- Effective-Max-Reservable-BW = max-reservable-bw (9Gbps) - dark-bw (2Gbps) = 7Gbps

In this example, the bandwidth available for RSVP-TE LSP admission is 7Gbps. This value is flooded in the network if the flooding threshold is crossed.

Note When you change the RSVP bandwidth percentage configuration or when the bundle capacity changes due to bundle-member state change, TE accounts for the dark bandwidth when new bandwidth values are advertised.

Note The measured dark bandwidth can be increased or decreased based on a configurable adjustment factor.

When the computed dark bandwidth increases for a link, it will lower the max-reservable-bw of that link, which might trigger preemption of the RSVP-TE LSPs. Preemption consists of freeing the resources of an established LSP, and assigning them to a new LSP. The freeing of resources causes a traffic disruption to the LSP that is being preempted.

Note SR-TE LSPs at the head-end are treated with the highest priority and cannot be preempted.

You can apply measured rates immediately using the **bandwidth-accounting apply all** command. When you apply measured rates immediately, the RSVP-TE bandwidth-accounting might flood the updated bandwidth values immediately. Applying measured rates immediately does not affect the periodic application of the bandwidth.

Configuring MPLS-TE

MPLS-TE requires co-ordination among several global neighbor routers. RSVP, MPLS-TE and IGP are configured on all routers and interfaces in the MPLS traffic engineering network. Explicit path and TE tunnel interfaces are configured only on the head-end routers. MPLS-TE requires some basic configuration tasks explained in this section.

Restrictions for MPLS-TE

- Dark bandwidth is not supported.

Building MPLS-TE Topology

Building MPLS-TE topology, sets up the environment for creating MPLS-TE tunnels. This procedure includes the basic node and interface configuration for enabling MPLS-TE. To perform constraint-based routing, you need to enable OSPF or IS-IS as IGP extension.

Before You Begin

Before you start to build the MPLS-TE topology, the following pre-requisites are required:

- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.
- Enable RSVP on the port interface.

Example

This example enables MPLS-TE on a node and then specifies the interface that is part of the MPLS-TE. Here, OSPF is used as the IGP extension protocol for information distribution.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface hundredGigE0/0/0/3
RP/0/RP0/CPU0:router(config)# router ospf area 1
RP/0/RP0/CPU0:router(config-ospf)# area 0
RP/0/RP0/CPU0:router(config-ospf-ar)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-ospf-ar)# interface hundredGigE0/0/0/3
RP/0/RP0/CPU0:router(config-ospf-ar-if)# exit
RP/0/RP0/CPU0:router(config-ospf)# mpls traffic-eng router-id 192.168.70.1
RP/0/RP0/CPU0:router(config)# commit
```

Example

This example enables MPLS-TE on a node and then specifies the interface that is part of the MPLS-TE. Here, IS-IS is used as the IGP extension protocol for information distribution.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface hundredGigE0/0/0/3
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# net 47.0001.0000.0000.0002.00
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide
RP/0/RP0/CPU0:router(config-isis-af)# mpls traffic-eng level 1
RP/0/RP0/CPU0:router(config-isis-af)# exit
```

```
RP/0/RP0/CPU0:router(config-isis)# interface hundredGigE0/0/0/3
RP/0/RP0/CPU0:router(config-isis-if)# exit
RP/0/RP0/CPU0:router(config)# commit
```

Related Topics

- How MPLS-TE Works, on page 83
- Creating an MPLS-TE Tunnel, on page 87

Creating an MPLS-TE Tunnel

Creating an MPLS-TE tunnel is a process of customizing the traffic engineering to fit your network topology. The MPLS-TE tunnel is created at the headend router. You need to specify the destination and path of the TE LSP.

To steer traffic through the tunnel, you can use the following ways:

- Static Routing
- Autoroute Announce
- Forwarding Adjacency

From the 7.1.1 release, IS-IS autoroute announce function is enhanced to redirect traffic from a source IP address prefix to a matching IP address assigned to an MPLS-TE tunnel destination interface.

Before You Begin

The following prerequisites are required to create an MPLS-TE tunnel:

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example

This example configures an MPLS-TE tunnel on the headend router with a destination IP address 192.168.92.125. The bandwidth for the tunnel, path-option, and forwarding parameters of the tunnel are also configured. You can use static routing, autoroute announce or forwarding adjacency to steer traffic through the tunnel.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# path-option 1 dynamic
RP/0/RP0/CPU0:router(config-if)# autoroute announce  or forwarding adjacency
RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 100
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the configuration of MPLS-TE tunnel using the following command.

```
RP/0/RP0/CPU0:router# show mpls traffic-engineering tunnels brief

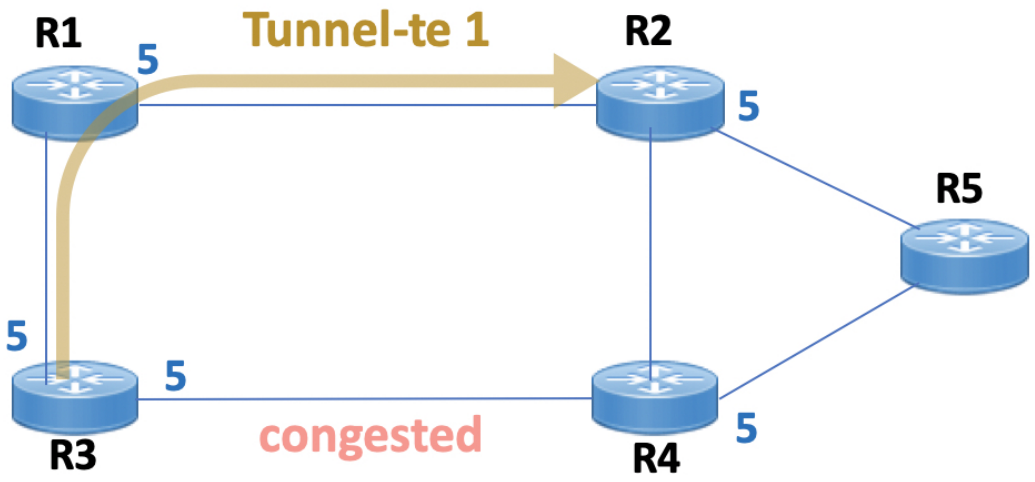
            Signalling Summary:
                LSP Tunnels Process:  running
                        RSVP Process:  running
                          Forwarding:  enabled
            Periodic reoptimization:  every 3600 seconds, next in 2538 seconds
             Periodic FRR Promotion:  every 300 seconds, next in 38 seconds
            Auto-bw enabled tunnels:  0 (disabled)
                       TUNNEL NAME         DESTINATION      STATUS  STATE


                      tunnel-te1       192.168.92.125      up       up
            Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

Automatic Modification Of An MPLS-TE Tunnel's Metric

If the IGP calculation on a router results in an equal cost multipath (ECMP) scenario where next-hop interfaces are a mix of MPLS-TE tunnels and physical interfaces, you may want to ensure that a TE tunnel is preferred. Consider this topology:

Figure 12: MPLS-TE Tunnel



1. All links in the network have a metric of 5.
2. To offload a congested link between R3 and R4, an MPLS-TE tunnel is created from R3 to R2.
3. If the metric of the tunnel is also 5, traffic from R3 to R5 is load-balanced between the tunnel and the physical R3-R4 link.

To ensure that the MPLS-TE tunnel is preferred in such scenarios, configure the **autoroute metric** command on the tunnel interface. The modified metric is applied in the routing information base (RIB), and the tunnel is preferred over the physical path of the same metric. Sample configuration:

```
Router# configure
Router(config)# interface tunnel-te 1
Router(config-if)# autoroute metric relative -1
```

The **autoroute metric** command syntax is **autoroute metric** {**absolute**|**relative**} *value*

- **absolute** enables the absolute metric mode, for a metric range between 1 and 2147483647.
- **relative** enables the relative metric mode, for a metric range between -10 and 10, including zero.

Note Since the **relative** metric is not saved in the IGP database, the advertised metric of the MPLS-TE tunnel remains 5, and doesn't affect SPF calculation outcomes on other nodes.

Related Topics

- How MPLS-TE Works, on page 83
- Building MPLS-TE Topology , on page 86

Teardown and Reestablishment of RSVP-TE Tunnels

Table 15: Feature History Table

Feature Name	Release Information	Feature Description
Teardown and Reestablishment of RSVP-TE Tunnels	Release 7.11.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]) You can now tear down and reestablish the existing tunnels of headend, midend, or tailend router tunnels of an MPLS network for optimized distribution of the traffic across MPLS and RSVP-TE to improve network performance and enhance resource utilization. Previously, you could reestablish tunnels only at the headend router using the mpls traffic-eng resetup command. The feature introduces these changes: **CLI**: mpls traffic-eng teardown **YANG Data Model**: Cisco-IOS-XR-mpls-te-act.yang (see GitHub, YANG Data Models Navigator)

In an MPLS-TE network which is configured with RSVP-TE, the headend, midend, and tailend router work together to establish and maintain tunnels or adjacencies for traffic engineering purposes. When the headend router boots up, it plays a critical role in MPLS-TE tunnel establishment using RSVP-TE signaling along the computed path with the tailend router.

During a system reboot, if a tailend router comes up first, the Interior Gateway Protocol (IGP) adjacency establishes a path for the tailend with the adjacent network devices. However, if the headend router comes up and starts creating tunnels during this tailend path creation, it may result in a poor distribution of tunnels at

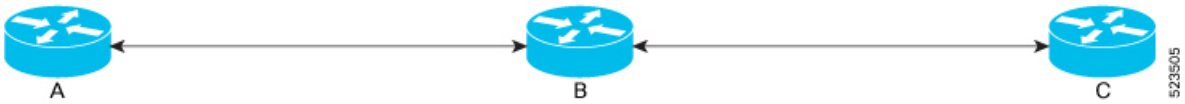
the tailend. In such cases, it's necessary to tear down all the tunnels and recreate them to ensure optimal distribution and functioning of the RSVP-TE as per the network conditions.

From Release 7.11.1, you can tear down and reestablish tunnels across all headend, midend, and tailend routers using the **mpls traffic-eng teardown** command with **all, head, mid, tail** parameters.

Table 16: Network Routers and Description

Router Name	Router Function	Tunnel Tear-Down and Reestablish Command
Headend Router	The headend router is responsible for determining the tunnel's path. It initiates the signaling process, specifying the tunnel's parameters and requirements. It sets up and manages the MPLS-TE tunnels.	**mpls traffic-eng teardown head**
Midend Router	The midend router, or an intermediate router or network device, is located between the headend and the tailend along the tunnel's path. It's involved in the signaling process and ensures the proper forwarding of traffic along the established tunnel.	**mpls traffic-eng teardown mid**
Tailend Router	The tailend router is the terminating router or network device located at the other end of the tunnel. It receives and processes the traffic that traverses through the tunnel and ensures proper traffic delivery to the destination.	**mpls traffic-eng teardown tail**

Figure 13: RSVP-TE tunnel Teardown and Reestablishment Topology



When RSVP-TE tunnel teardown message is triggered and:

- Router B is configured as tailend and Router A is configured as headend: A **ResvTear upstream** message is sent to the headend router. This message informs the headend router to tear down the RSVP-TE tunnel and release the resources associated with the tunnel.
- Router B is configured as headend and Router C as tailend: A **PathTear downstream** message is sent to the tailend router. A headend router triggers the process of recomputing the tunnels. The headend router (Router B) initiates the process of recomputing the tunnels, which involves recalculating the path and parameters for establishing new tunnels.
- Router B is configured as midend where Router A and Router B as headend and tailend respectively: A **ResvTear upstream** message is sent to the headend router and a **PathTear downstream** message is sent to the tailend router. These messages inform the respective routers to tear down the RSVP-TE tunnel and release the associated resources.

You can tear down and set up all types of tunnels including P2P, P2MP, numbered, named, and auto tunnels.

Limitations

- A maximum 90 seconds are required for the tunnels to get reestablished once they are torn.
- Use the **mpls traffic-eng resetup** command to reestablish the tunnels only at the headend router.

Configure Tear down and Reestablishment of RSVP-TE Tunnels

Configuration Example

Use the **mpls traffic-eng teardown all** command to tear down and reestablish all the RSVP-TE tunnels in a network node. This command must be executed in XR EXEC mode.

```
Router# mpls traffic-eng teardown all
```

Use the **mpls traffic-eng teardown head** command to tear down and reestablish the RSVP-TE tunnels at the headend router. This command must be executed in XR EXEC mode.

```
Router# mpls traffic-eng teardown head
```



Note You can also use the **mpls traffic-eng resetup** command to reestablish tunnels only at the headend router in XR EXEC mode.

Use the **mpls traffic-eng teardown mid** command to tear down and reestablish the RSVP-TE tunnels at the midend router. This command must be executed in XR EXEC mode.

```
Router# mpls traffic-eng teardown mid
```

Use the **mpls traffic-eng teardown tail** command to tear down and reestablish the RSVP-TE tunnels at the tailend router. This command must be executed in XR EXEC mode.

```
Router# mpls traffic-eng teardown tail
```

Use the **show mpls traffic-eng tunnels summary** command to check RSVP-TE tunnel status after you run the teardown command.

```
Router# show mpls traffic-eng tunnels summary
Output received:
Thu Sep 14 10:48:45.007 UTC

        Path Selection Tiebreaker:  Min-fill (default)
              LSP Tunnels Process:  running
                     RSVP Process:  running
                       Forwarding:  enabled
          Periodic reoptimization:  every 3600 seconds, next in 2806 seconds
           Periodic FRR Promotion:  every 300 seconds, next in 9 seconds
      Periodic auto-bw collection:  5 minute(s) (disabled)

Signalling Summary:
    Head: 14006 interfaces, 14006 active signalling attempts, 14006 established
          14006 explicit, 0 dynamic
          14006 activations, 0 deactivations
          0 recovering, 0 recovered
    Mids: 2000
    Tails: 4003

Fast ReRoute Summary:
    Head:     14000 FRR tunnels, 14000 protected, 0 rerouted
    Mid:      2000 FRR tunnels, 2000 protected, 0 rerouted
```

```
    Summary:  16000 protected, 13500 link protected, 2500 node protected, 0 bw protected
    Backup:   6 tunnels, 4 assigned
    Interface: 10 protected, 0 rerouted

Bidirectional Tunnel Summary:
    Tunnel Head: 0 total, 0 connected, 0 associated, 0 co-routed
    Tunnel Tail: 0 total, 0 connected, 0 associated, 0 co-routed
    LSPs Head:   0 established, 0 proceeding, 0 associated, 0 standby
    LSPs Mid:    0 established, 0 proceeding, 0 associated, 0 standby
    LSPs Tail:   0 established, 0 proceeding, 0 associated, 0 standby
```

Configuring Fast Reroute

Fast reroute (FRR) provides link protection to LSPs enabling the traffic carried by LSPs that encounter a failed link to be rerouted around the failure. The reroute decision is controlled locally by the router connected to the failed link. The headend router on the tunnel is notified of the link failure through IGP or through RSVP. When it is notified of a link failure, the headend router attempts to establish a new LSP that bypasses the failure. This provides a path to reestablish links that fail, providing protection to data transfer. The path of the backup tunnel can be an IP explicit path, a dynamically calculated path, or a semi-dynamic path. For detailed conceptual information on fast reroute, see MPLS-TE Features - Details, on page 132

Before You Begin

The following prerequisites are required to create an MPLS-TE tunnel:

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example

This example configures fast reroute on an MPLS-TE tunnel. Here, tunnel-te 2 is configured as the back-up tunnel. You can use the **protected-by** command to configure path protection for an explicit path that is protected by another path.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# fast-reroute
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# backup-path tunnel-te 2
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# backup-bw global-pool 5000
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125
RP/0/RP0/CPU0:router(config-if)# path-option l explicit name backup-path protected by 10
RP/0/RP0/CPU0:router(config-if)# path-option l0 dynamic
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Use the **show mpls traffic-eng fast-reroute database** command to verify the fast reroute configuration.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng fast-reroute database
```

```
Tunnel head FRR information:
Tunnel      Out intf/label                    FRR intf/label      Status
----------  ----------------                  ----------------    -------
tt4000       HundredGigabitEthernet 0/0/0/3:34    tt1000:34        Ready
tt4001       HundredGigabitEthernet 0/0/0/3:35    tt1001:35        Ready
tt4002       HundredGigabitEthernet 0/0/0/3:36    tt1001:36        Ready
```

Related Topics

- Configuring MPLS-TE, on page 86
- Configuring Auto-Tunnel Backup, on page 93
- Configuring Next Hop Backup Tunnel, on page 96
- MPLS-TE Features - Details, on page 132

Configuring Auto-Tunnel Backup

Table 17: Feature History Table

Feature Name	Release Information	Feature Description
Bandwidth Protection Functions to Enhance auto-tunnel backup Capabilities	Release 7.5.1	This feature introduces bandwidth protection functions for auto-tunnel backups, such as signaled bandwidth, bandwidth protection, and soft-preemption. These functions provide better bandwidth usage and prevent traffic congestion and traffic loss. In earlier releases, auto-tunnel backups provided only link protection and node protection. Backup tunnels were signaled with zero bandwidth, causing traffic congestion when FRR went active. This feature introduces the following commands and keywords: • **bandwidth-protection maximum-aggregate** • **signalled-bandwidth** • **soft-preemption**

The MPLS Traffic Engineering Auto-Tunnel Backup feature enables a router to dynamically build backup tunnels on the interfaces that are configured with MPLS TE tunnels instead of building MPLS-TE tunnels statically.

The MPLS-TE Auto-Tunnel Backup feature has these benefits:

- Backup tunnels are built automatically, eliminating the need for users to pre-configure each backup tunnel and then assign the backup tunnel to the protected interface.
- Protection is expanded—FRR does not protect IP traffic that is not using the TE tunnel or Label Distribution Protocol (LDP) labels that are not using the TE tunnel.

The TE attribute-set template that specifies a set of TE tunnel attributes, is locally configured at the headend of auto-tunnels. The control plane triggers the automatic provisioning of a corresponding TE tunnel, whose characteristics are specified in the respective attribute-set.

Configuration Example

This example configures Auto-Tunnel backup on an interface and specifies the attribute-set template for the auto tunnels. In this example, unused backup tunnels are removed every 20 minutes using a timer and also the range of tunnel interface numbers are specified.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# auto-tunnel backup
RP/0/RP0/CPU0:router(config-mpls-te-if-auto-backup)# attribute-set ab
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup timers removal unused 20
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup tunnel-id min 6000 max 6500
RP/0/RP0/CPU0:router(config)# commit
```

Verification

This example shows a sample output for automatic backup tunnel configuration.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels brief

             TUNNEL NAME          DESTINATION      STATUS  STATE
              tunnel-te0            200.0.0.3          up  up
              tunnel-te1            200.0.0.3          up  up
              tunnel-te2            200.0.0.3          up  up
              tunnel-te50           200.0.0.3          up  up
            *tunnel-te60         200.0.0.3           up  up
               *tunnel-te70         200.0.0.3          up  up
               *tunnel-te80         200.0.0.3          up  up
```

Related Topics

- Configuring Fast Reroute , on page 92
- Configuring Next Hop Backup Tunnel, on page 96
- MPLS-TE Features - Details, on page 132

Bandwidth Protection Functions to Enhance auto-tunnel backup Capabilities

Without bandwidth protection, auto-tunnel backups provide only link protection and node protection (per next-next-hop), and backup tunnels are signalled with zero bandwidth. This causes traffic congestion when FRR goes active, since the backup tunnels might be protecting huge amount of data, such as LSPs with large bandwidth or multiple LSPs.

To address the congestion issue, bandwidth protection capabilities are added for auto-tunnel backups. Bandwidth protection, signalled bandwidth, and soft-preemption settings are provided. Details:

- *Bandwidth protection* – A link or node protection backup might not provide bandwidth protection. But with this setting (**bandwidth-protection maximum-aggregate**), you can set the maximum bandwidth value that an auto-tunnel can protect.
- *Signalled bandwidth* – Without bandwidth protection, auto-tunnel backups are signaled with zero bandwidth too, with no guarantee that at least some bandwidth is backed up. So, the backup tunnels might be setup on links that are highly utilized, causing congestion drops when the backup tunnels start to transmit traffic after FRR is triggered.

  This setting (**signalled-bandwidth**) addresses the issue, since you can set the signalled bandwidth of the tunnel (and reserve minimal bandwidth for an auto-tunnel backup). When you set the signal bandwidth value for auto-backup tunnels, congestion over backup links reduces.
- *Soft-preemption* – Since bandwidth can be reserved for autobackup tunnels, a setting (**soft-preemption**) is provided for soft-preemption of the reserved bandwidth, if it is needed for a higher-priority tunnel.

Configurations

```
/*Enable Bandwidth Protection On a TE Auto-Tunnel Backup*/

Router # configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# interface GigabitEthernet 0/2/0/0 auto-tunnel backup
Router(config-te-if-auto-backup)# bandwidth-protection maximum-aggregate 100000
Router(config-te-if-auto-backup)# commit

/*Enable Signalled Bandwidth On a TE Auto-Tunnel Backup*/

Router # configure
Router(config)# mpls traffic-eng attribute-set auto-backup MyBackupConfig
Router(config-te-attribute-set)# signalled-bandwidth 700000
Router(config-te-attribute-set)# commit
```

After creating the auto backup attribute-set (**MyBackupConfig** in this case), associate with the auto-tunnel backup interface.

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# interface GigabitEthernet 0/2/0/0 auto-tunnel backup
Router(config-te-if-auto-backup)# attribute-set MyBackupConfig
Router(config-te-if-auto-backup)# auto-tunnel backup tunnel-id min 6000 max 6500
Router(config-mpls-te)# commit

/*Enable Soft-Preemption Bandwidth On a TE Auto-Tunnel Backup*/

Router# configure
Router(config)# mpls traffic-eng attribute-set auto-backup MyBackupConfig
Router(config-te-attribute-set)# soft-preemption
Router(config-te-attribute-set)# commit
```

Verification

```
/*Verify Auto-Tunnel Backup Configuration*/
```

In the output, bandwidth protection details are displayed, as denoted by *BW*.

```
Router# show mpls traffic-eng auto-tunnel backup

AutoTunnel Backup Configuration:
    Interfaces count: 1
    Unused removal timeout: 1h 0m 0s
    Configured tunnel number range: 6000-6500
```

```
AutoTunnel Backup Summary:
    AutoTunnel Backups:
             0 created, 0 up, 0 down, 0 unused
            0 NHOP, 0 NNHOP, 0 SRLG strict, 0 SRLG preferred, 0 SRLG weighted, 0 BW protected

  Protected LSPs:
             0 NHOP, 0 NHOP+SRLG, 0 NHOP+BW, 0 NHOP+BW+SRLG
             0 NNHOP, 0 NNHOP+SRLG, 0 NNHOP+BW, 0 NNHOP+BW+SRLG
    Protected S2L Sharing Families:
             0 NHOP, 0 NHOP+SRLG, 0 NNHOP+BW, 0 NNHOP+BW+SRLG
             0 NNHOP, 0 NNHOP+SRLG, 0 NNHOP+BW, 0 NNHOP+BW+SRLG
    Protected S2Ls:
             0 NHOP, 0 NHOP+SRLG, 0 NNHOP+BW, 0 NNHOP+BW+SRLG
             0 NNHOP, 0 NNHOP+SRLG, 0 NNHOP+BW, 0 NNHOP+BW+SRLG

 Cumulative Counters (last cleared 00:08:47 ago):
                         Total   NHOP  NNHOP
    Created:               0       0      0
    Connected:             0       0      0
    Removed (down):        0       0      0
    Removed (unused):      0       0      0
    Removed (in use):      0       0      0
    Range exceeded:        0       0      0
```

Configuring Next Hop Backup Tunnel

The backup tunnels that bypass only a single link of the LSP path are referred as Next Hop (NHOP) backup tunnels because they terminate at the LSP's next hop beyond the point of failure. They protect LSPs, if a link along their path fails, by rerouting the LSP traffic to the next hop, thus bypassing the failed link.

Configuration Example

This example configures next hop backup tunnel on an interface and specifies the attribute-set template for the auto tunnels. In this example, unused backup tunnels are removed every 20 minutes using a timer and also the range of tunnel interface numbers are specified.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# auto-tunnel backup nhop-only
RP/0/RP0/CPU0:router(config-mpls-te-if-auto-backup)# attribute-set ab
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup timers removal unused 20
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup tunnel-id min 6000 max 6500
RP/0/RP0/CPU0:router(config)# commit
```

Related Topics

- Configuring Auto-Tunnel Backup, on page 93
- Configuring Fast Reroute , on page 92
- MPLS-TE Features - Details, on page 132

Configuring SRLG Node Protection

Shared Risk Link Groups (SRLG) in MPLS traffic engineering refer to situations in which links in a network share common resources. These links have a shared risk, and that is when one link fails, other links in the group might fail too.

OSPF and IS-IS flood the SRLG value information (including other TE link attributes such as bandwidth availability and affinity) using a sub-type length value (sub-TLV), so that all routers in the network have the SRLG information for each link.

MPLS-TE SRLG feature enhances backup tunnel path selection by avoiding using links that are in the same SRLG as the interfaces it is protecting while creating backup tunnels.

Configuration Example

This example creates a backup tunnel and excludes the protected node IP address from the explicit path.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# backup-path tunnl-te 2
RP/0/RP0/CPU0:router(config-mpls-te-if)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# path-option 1 explicit name backup-srlg
RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# explicit-path name backup-srlg-nodep
RP/0/RP0/CPU0:router(config-if)# index 1 exclude-address 192.168.91.1
RP/0/RP0/CPU0:router(config-if)# index 2 exclude-srlg 192.168.92.2
RP/0/RP0/CPU0:router(config)# commit
```

Related Topics

- Configuring Fast Reroute , on page 92
- MPLS-TE Features - Details, on page 132

Configuring Pre-Standard DS-TE

Regular traffic engineering does not provide bandwidth guarantees to different traffic classes. A single bandwidth constraint is used in regular TE that is shared by all traffic. MPLS DS-TE enables you to configure multiple bandwidth constraints on an MPLS-enabled interface. These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint. Cisco IOS XR software supports two DS-TE modes: Pre-standard and IETF. Pre-standard DS-TE uses the Cisco proprietary mechanisms for RSVP signaling and IGP advertisements. This DS-TE mode does not interoperate with third-party vendor equipment. Pre-standard DS-TE is enabled only after configuring the sub-pool bandwidth values on MPLS-enabled interfaces.

Pre-standard Diff-Serve TE mode supports a single bandwidth constraint model a Russian Doll Model (RDM) with two bandwidth pools: global-pool and sub-pool.

Before You Begin

The following prerequisites are required to configure a Pre-standard DS-TE tunnel.

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example

This example configures a pre-standard DS-TE tunnel.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth 100 150 sub-pool 50
RP/0/RP0/CPU0:router(config-rsvp-if)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# signalled bandwidth sub-pool 10
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Use the **show mpls traffic-eng topology** command to verify the pre-standard DS-TE tunnel configuration.

Related Topics

- Configuring an IETF DS-TE Tunnel Using RDM, on page 98
- Configuring an IETF DS-TE Tunnel Using MAM, on page 99
- MPLS-TE Features - Details, on page 132

Configuring an IETF DS-TE Tunnel Using RDM

IETF DS-TE mode uses IETF-defined extensions for RSVP and IGP. This mode interoperate with third-party vendor equipment.

IETF mode supports multiple bandwidth constraint models, including Russian Doll Model (RDM) and Maximum Allocation Model (MAM), both with two bandwidth pools. In an IETF DS-TE network, identical bandwidth constraint models must be configured on all nodes.

Before you Begin

The following prerequisites are required to create a IETF mode DS-TE tunnel using RDM:

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example

This example configures an IETF DS-TE tunnel using RDM.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth rdm 100 150
RP/0/RP0/CPU0:router(config-rsvp-if)# exit
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# ds-te mode ietf
RP/0/RP0/CPU0:router(config-mpls-te)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# signalled bandwidth sub-pool 10 class-type 1
```

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

Verification

Use the **show mpls traffic-eng topology** command to verify the IETF DS-TE tunnel using RDM configuration.

Related Topics

- Configuring Pre-Standard DS-TE, on page 97
- Configuring an IETF DS-TE Tunnel Using MAM, on page 99
- MPLS-TE Features - Details, on page 132

Configuring an IETF DS-TE Tunnel Using MAM

IETF DS-TE mode uses IETF-defined extensions for RSVP and IGP. This mode interoperates with third-party vendor equipment. IETF mode supports multiple bandwidth constraint models, including Russian Doll Model (RDM) and Maximum Allocation Model (MAM), both with two bandwidth pools.

Configuration Example

This example configures an IETF DS-TE tunnel using MAM.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth mam max-reservable-bw 1000 bc0 600 bc1 400
RP/0/RP0/CPU0:router(config-rsvp-if)# exit
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# ds-te mode ietf
RP/0/RP0/CPU0:router(config-mpls-te)# ds-te bc-model mam
RP/0/RP0/CPU0:router(config-mpls-te)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# signalled bandwidth sub-pool 10
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Use the **show mpls traffic-eng topology** command to verify the IETF DS-TE tunnel using MAM configuration.

Related Topics

- Configuring an IETF DS-TE Tunnel Using RDM, on page 98
- Configuring Pre-Standard DS-TE, on page 97
- MPLS-TE Features - Details, on page 132

Configuring Flexible Name-Based Tunnel Constraints

MPLS-TE Flexible Name-based Tunnel Constraints provides a simplified and more flexible means of configuring link attributes and path affinities to compute paths for the MPLS-TE tunnels.

In traditional TE, links are configured with attribute-flags that are flooded with TE link-state parameters using Interior Gateway Protocols (IGPs), such as Open Shortest Path First (OSPF).

MPLS-TE Flexible Name-based Tunnel Constraints lets you assign, or map, up to 32 color names for affinity and attribute-flag attributes instead of 32-bit hexadecimal numbers. After mappings are defined, the attributes can be referred to by the corresponding color name.

Configuration Example

This example shows assigning a how to associate a tunnel with affinity constraints.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# affinity-map red 1
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# attribute-names red
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# affinity include red
RP/0/RP0/CPU0:router(config)# commit
```

Configuring Automatic Bandwidth

Automatic bandwidth allows you to dynamically adjust bandwidth reservation based on measured traffic. MPLS-TE automatic bandwidth monitors the traffic rate on a tunnel interface and resizes the bandwidth on the tunnel interface to align it closely with the traffic in the tunnel. MPLS-TE automatic bandwidth is configured on individual Label Switched Paths (LSPs) at every headend router.

The following table specifies the parameters that can be configured as part of automatic bandwidth configuration.

Table 18: Automatic Bandwidth Parameters

Bandwidth Parameters	Description
Application frequency	Configures how often the tunnel bandwidths changed for each tunnel. The default value is 24 hours.
Bandwidth limit	Configures the minimum and maximum automatic bandwidth to set on a tunnel.
Bandwidth collection frequency	Enables bandwidth collection without adjusting the automatic bandwidth. The default value is 5 minutes.
Overflow threshold	Configures tunnel overflow detection.
Adjustment threshold	Configures the tunnel-bandwidth change threshold to trigger an adjustment.

Configuration Example

This example enables automatic bandwidth on MPLS-TE tunnel interface and configure the following automatic bandwidth variables.

- Application frequency
- Bandwidth limit
- Adjustment threshold

- Overflow detection

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# auto-bw
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# application 1000
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# bw-limit min 30 max 1000
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# adjustment-threshold 50 min 800
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# overflow threshold 100 limit 1
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the automatic bandwidth configuration using the **show mpls traffic-eng tunnels auto-bw brief** command.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels auto-bw brief

  Tunnel    LSP  Last appl  Requested  Signalled    Highest    Application
  Name       ID   BW(kbps)   BW(kbps)   BW(kbps)   BW(kbps)      Time Left
  -------------- ------ ---------- ---------- ---------- ---------- --------------
  tunnel-te1      5          500         300        420          1h 10m
```

Related Topics

- MPLS-TE Features - Details, on page 132

Enable per-application accounting for interfaces

The following configurations help enable per-application accounting for MPLS RSVP-TE IPv4 traffic and segment routing with MPLS IPv4 traffic on the interfaces respectively.

Example 1: Enable per-application accounting for MPLS RSVP-TE IPv4 traffic on the interfaces

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# accounting
RP/0/RP0/CPU0:router(config-acct)# interfaces
RP/0/RP0/CPU0:router(config-acct-if)# mpls
RP/0/RP0/CPU0:router(config-acct-if-mpls)# ipv4 rspv-te
```

Example 2: Enable per-application accounting for segment routing MPLS IPv4 traffic on the interfaces

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# accounting
RP/0/RP0/CPU0:router(config-acct)# interfaces
RP/0/RP0/CPU0:router (config-acct-if)# segment-routing mpls ipv4
```

Note The dark bandwidth accounting feature is not supported for segment routing with MPLS IPv6 traffic on the router.

Configure RSVP-TE dark bandwidth accounting

Perform these steps to enable RSVP-TE bandwidth accounting and dark bandwidth advertisement for all MPLS-TE enabled links:

Before you begin

Make sure that the traffic accounting feature is enabled on the router.

Procedure

Step 1 **configure**

Example:

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

Enters global configuration mode.

Step 2 **mpls traffic-eng**

Example:

```
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
```

Enters MPLS TE configuration mode.

Step 3 **bandwidth-accounting**

Example:

```
RP/0/RP0/CPU0:router(config-mpls-te)# bandwidth-accounting
```

Enables RSVP-TE dark bandwidth accounting and enters bandwidth accounting configuration mode.

Step 4 **application interval** *seconds*

Example:

```
RP/0/RP0/CPU0:router(config-mpls-te-bw-account)# application interval 90
```

Configures the length of the application interval in seconds. At the end of application interval, dark bandwidth rates are computed and applied to all RSVP-TE enabled interfaces.

Note
Model-driven telemetry supports dark bandwidth. The telemetry polling interval is reduced to 10 seconds.

If the interval is reconfigured while the timer is running, the new value is compared to the time remaining for the running timer. The timer is adjusted so that the lower of these two values is used for this interval. The subsequent interval will use the newly configured value.

Note
TE stores sample history for the current and previous application intervals. If the application interval is lowered, TE may discard the sample history.

Range is from 90 to 1800. The default value is 180.

Step 5 **application enforced**

Example:

```
RP/0/RP0/CPU0:router(config-mpls-te-bw-account)# application enforced
```

Enables enforcement of the calculated effective maximum reservable bandwidth rate. This configuration is mandatory for the RSVP dark bandwidth accounting feature to work properly.

Step 6 **adjustment-factor** *percentage*

Example:

```
RP/0/RP0/CPU0:router(config-mpls-te-bw-account)# adjustment-factor 85
```

Configures TE to over-book (>100%) or under-book (<100%) the effective maximum reservable bandwidth. The measured dark-bandwidth will be scaled based on the adjustment factor. Range is from 0 to 200. The default value is 100.

Step 7 **sampling-interval** *seconds*

Example:

```
RP/0/RP0/CPU0:router(config-mpls-te-bw-account)# sampling-interval 30
```

Configures the length of the sampling interval in seconds. The bandwidth rate is collected from the statistics collector process (statsD) at the end of each sampling interval for each TE link.

If the interval is reconfigured while the timer is running, the new value is compared to the time remaining for the running timer. The timer is adjusted so that the lower of these two values is used for this interval. The subsequent interval will use the newly configured value.

Range is from 30 to 600. The default is 60.

Step 8 **flooding threshold** { **up** | **down** } *percentage*

Example:

```
RP/0/RP0/CPU0:router(config-mpls-te-bw-account)# flooding threshold up 30 down 30
```

Configures the reserved bandwidth thresholds. When bandwidth crosses one of these thresholds, flooding is triggered. Range is from 0 to 100. The default value is 10.

Step 9 **commit**

Step 10 **show mpls traffic-eng link-management summary**

Example:

```
RP/0/RP0/CPU0:router# show mpls traffic-eng link-management summary
```

(Optional)

Displays a summary of link management information, including bandwidth accounting information.

Step 11 **show mpls traffic-eng link-management advertisements**

Example:

```
RP/0/RP0/CPU0:router# show mpls traffic-eng link-management advertisements
```

(Optional)

Displays local link information that MPLS-TE link management is currently flooding into the global TE topology.

Step 12 **show mpls traffic-eng link-management interfaces** [*type interface-path-id*]

Example:

```
RP/0/RP0/CPU0:router# show mpls traffic-eng link-management interfaces gig0/1/1/1 detail
```

(Optional)

Displays bandwidth accounting and utilization details and link management information.

Autoroute Announce

The segment routing tunnel can be advertised into an Interior Gateway Protocol (IGP) as a next hop by configuring the autoroute announce statement on the source router. The IGP then installs routes in the Routing Information Base (RIB) for shortest paths that involve the tunnel destination. Autoroute announcement of IPv4 prefixes can be carried through either OSPF or IS-IS. Autoroute announcement of IPv6 prefixes can be carried only through IS-IS. IPv6 forwarding over tunnel needs additional parameter in tunnel configuration.

Restrictions

The Autoroute Announce feature is supported with the following restrictions:

- A maximum on 128 tunnels can be announced.
- Tunnels in ECMP may not be path diverse.

Note Configuring Segment Routing and Autoroute Destination together is not supported. If autoroute functionality is required in an Segment Routing network, we recommend you to configure Autoroute Announce.

Configuration

To specify that the Interior Gateway Protocol (IGP) should use the tunnel (if the tunnel is up) in its enhanced shortest path first (SPF) calculation, use the **autoroute announce** command in interface configuration mode.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 10
RP/0/RP0/CPU0:router(config-if)# autoroute announce [include-ipv6]
RP/0/RP0/CPU0:router(config-if)# commit
```

Verification

Verify the route using the following commands:

```
RP/0/RP0/CPU0:router# Show route
Mon Aug  8 00:31:29.406 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, (!) - FRR Backup path

Gateway of last resort is not set

i L1 9.1.0.0/24 [115/30] via 9.9.9.9, 2d19h, tunnel-te3195
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3196
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3197
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3198
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3199
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3200
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3201
                [115/30] via 9.9.9.9, 2d19h, tunnel-te3202
i L1 9.9.9.9/32 [115/30] via 9.9.9.9, 2d22h, tunnel-te3195
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3196
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3197
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3198
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3199
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3200
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3201
                [115/30] via 9.9.9.9, 2d22h, tunnel-te3202

RP/0/RP0/CPU0:router# show route ipv6
Tue Apr 26 04:02:07.968 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, (!) - FRR Backup path

Gateway of last resort is not set

L    ::ffff:127.0.0.0/104
      [0/0] via ::, 00:17:36
C    2002::/64 is directly connected,
      00:39:01, TenGigE0/0/0/2
L    2002::1/128 is directly connected,
      00:39:01, TenGigE0/0/0/2
i L1 2003::/64
      [115/20] via ::, 00:02:32, tunnel-te992
      [115/20] via ::, 00:02:32, tunnel-te993
      [115/20] via ::, 00:02:32, tunnel-te994
      [115/20] via ::, 00:02:32, tunnel-te995
      [115/20] via ::, 00:02:32, tunnel-te996
      [115/20] via ::, 00:02:32, tunnel-te997
      [115/20] via ::, 00:02:32, tunnel-te998
      [115/20] via ::, 00:02:32, tunnel-te999
C    2006::/64 is directly connected,
      00:39:01, TenGigE0/0/0/6
L    2006::1/128 is directly connected,
```

```
      00:39:01, TenGigE0/0/0/6
i L1 2007::/64
      [115/20] via ::, 00:02:32, tunnel-te992
      [115/20] via ::, 00:02:32, tunnel-te993
      [115/20] via ::, 00:02:32, tunnel-te994
      [115/20] via ::, 00:02:32, tunnel-te995
      [115/20] via ::, 00:02:32, tunnel-te996
      [115/20] via ::, 00:02:32, tunnel-te997
      [115/20] via ::, 00:02:32, tunnel-te998
      [115/20] via ::, 00:02:32, tunnel-te999
```

```
show cef ipv6 2007::5 hardware egress location 0/0/CPU0
Sun May  1 03:18:55.151 UTC 2007::/64, version 50, internal 0x1000001 0x0 (ptr 0x895e819c)
 [1], 0x0 (0x894d3678), 0x0 (0x0)  Updated May  1 03:13:49.066  Prefix Len 64, traffic index
 0, precedence n/a, priority 2
   via ::/128, tunnel-te1, 2 dependencies, weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0x8b5ea420 0x0]
    next hop ::/128
    local adjacency
   via ::/128, tunnel-te2, 2 dependencies, weight 0, class 0 [flags 0x0]
    path-idx 1 NHID 0x0 [0x8b5ea5e0 0x0]
    next hop ::/128
    local adjacency


 LEAF - HAL pd context :
 sub-type : IPV6, ecd_marked:0, has_collapsed_ldi:0  collapse_bwalk_required:0, ecdv2_marked:0
 HW Walk:
LEAF:
    Handle: 0xaabbccdd type: 1 FEC handle: 0x8999fb18

    LWLDI:
        PI:0x894d3678 PD:0x894d36b8 rev:214 p-rev:212 ldi type:3
        FEC hdl: 0x8999fb18 fec index: 0x0(0) num paths:2, bkup: 0

        SHLDI:
            PI:0x893658e8 PD:0x89365968 rev:212 p-rev:0 flag:0x0
            FEC hdl: 0x8999fb18 fec index: 0x20000002(2) num paths: 2 bkup paths: 2
            Path:0 fec index: 0x20004000(16384) DSP:0x440 Dest fec index: 0x2000101c(4124)
            Path:1 fec index: 0x20004002(16386) DSP:0x440 Dest fec index: 0x2000101c(4124)
            Path:2 fec index: 0x20004001(16385) DSP:0x440 Dest fec index: 0x2000101c(4124)
            Path:3 fec index: 0x20004003(16387) DSP:0x440 Dest fec index: 0x2000101c(4124)

V6TE NH HAL PD context :
pdptr 0x8b5ea488, flags :1, index:0

              TE-NH:
              Flag: PHP HW Tun_O_Tun frr_active_chg te_protect_chg , PD(0x880d1640), Push:
 0, Swap: 0, Link: 0
            ifhandle:0x800001c llabel:24001 FEC hdl: 0x896d4eb8 fec index: 0x20001022(4130),
 SRTE fec hdl:(nil)

V6TE NH HAL PD context :
pdptr 0x8b5ea648, flags :1, index:0

              TE-NH:
              Flag: PHP HW , PD(0x880d12d0), Push: 0, Swap: 0, Link: 0x8b34d518
            ifhandle:0x8000024 llabel:24000 FEC hdl: 0x896d4238 fec index: 0x2000101c(4124),
 SRTE fec hdl:(nil)
```

Autoroute announce with IS-IS

Autoroute Announce (AA) with IS-IS is a feature that allows routers to install MPLS-TE tunnels to destinations based on the shortest path calculated by IS-IS, improving traffic engineering and path selection for anycast prefixes.

Table 19: Feature History Table

Feature Name	Release	Feature Description
MPLS-TE IPv6-only autoroute announce	Release 25.4.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]) This feature allows you to disable IPv4 autoroute announce without turning off autoroute announce entirely. To achieve IPv6-only announcements over MPLS-TE tunnels, use the new **exclude-ipv4** option along with the **include-ipv6** option in the autoroute announce configuration. This feature introduces these changes: CLI: A new keyword, **exclude-ipv4**, has been added to the **autoroute announce** command.

Feature Name	Release	Feature Description
Autoroute announce with IPv6 destination address for IS-IS	Release 25.2.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]) The autoroute announce feature now allows an MPLS-TE tunnel as the nexthop for IPv6 destinations thus ensuring a seamless IPv6 traffic engineering. IS-IS uses this IPv6 destination address when filtering prefixes. This feature introduces these changes: CLI: Two new keywords, **destination** and **ipv6**, were introduced in the **autoroute announce** command.

Feature Name	Release	Feature Description
Autoroute announce with IS-IS for anycast prefixes	Release 7.5.4	We have enabled seamless migration from autoroute announce (AA) tunnels with OSPF to AA tunnels with IS-IS. This is possible because you can now use AA tunnels with IS-IS to calculate the underlying native IGP metric for anycast prefixes to select the shortest path and then select a tunnel on that shortest path. Previously, autoroute announce tunnels with IS-IS behaved differently as compared to OSPF. IS-IS used the autoroute announce metric whereas, OSPF uses the underlying native IGP cost. With this feature the router uses the same method of cost calculation in autoroute announce in both OSPF and IS-IS. This feature introduces these: • CLI: **anycast-prefer-igp-cost** • YANG Data model: New Xpaths for Cisco-IOS-XR-um-router-isis-cfg.yang and Cisco-IOS-XR-clns-isis-cfg.yang (see GitHub and Yang Data Navigator)

When you configure Autoroute Announce (AA) tunnels, IGP installs the tunnel to the destination in the Routing Information Base (RIB) for the shortest paths. However, the behavior of AA for anycast destination is different from IS-IS compared to OSPF.

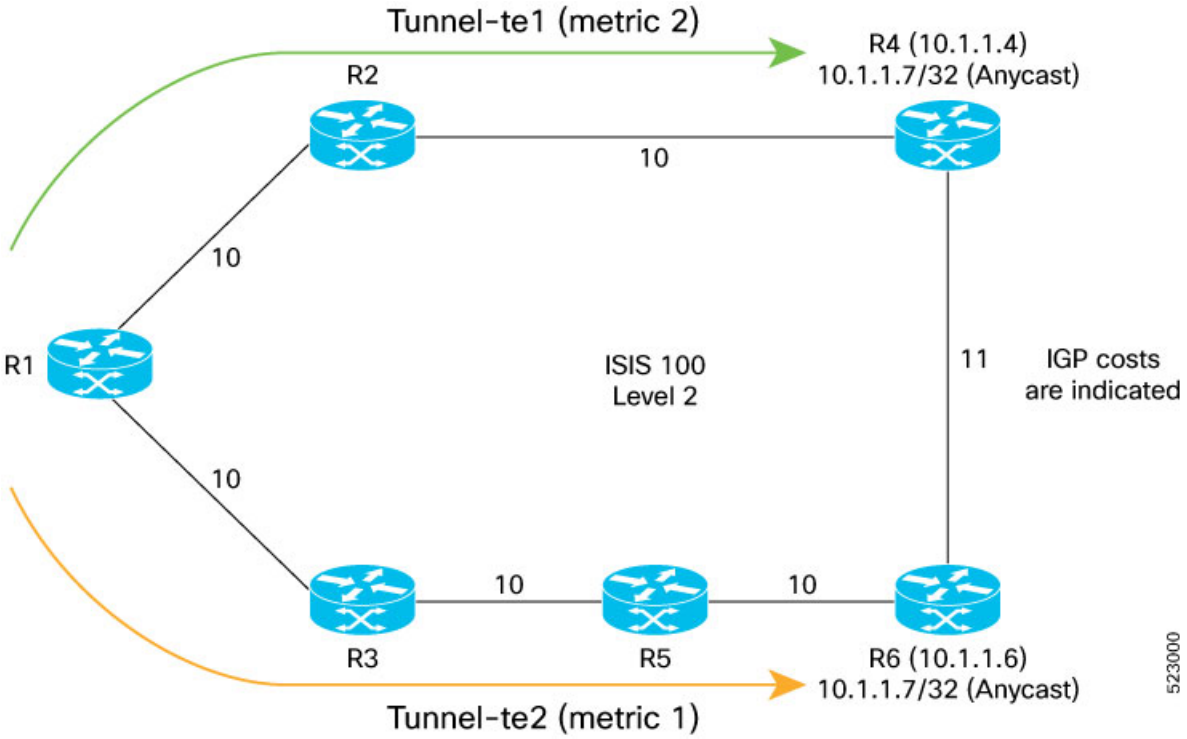
- IS-IS: AA tunnel with IS-IS considers the AA metric to calculate the prefix reachability for the shortest path.
- OSPF: AA with OSPF uses the lowest IGP cost to find the shortest path. And, then use the TE tunnel which is on that shortest path.

We now introduce the **anycast-prefer-igp-cost** command for AA tunnel with IS-IS anycast prefixes to allow the router to choose the shortest IGP path and select the tunnel on that shortest path, similar to OSPF behavior.

Starting from Cisco IOS XR Release 25.4.1, the autoroute announce feature restricts the usage of MPLS-TE tunnels by IGP to IPv6 prefixes. To enable the IPv6-only autoroute announce feature, configure the **autoroute announce exclude-ipv4** and **autoroute announce include-ipv6** commands.

Topology

Using this topology, let's see how the shortest path is chosen based on the IGP cost:



- In this topology, R1 is the headend and has MPLS RSVP-TE tunnels to R2 and R3.
- The tunnel destinations are loopback addresses of R4 and R6, which is 10.1.1.4 and 10.1.1.6.
- R1 uses AA to allow traffic to prefixes advertised by R2 and R3 to go over the MPLS-TE tunnel.
- Two AA tunnels are configured with different metrics and destinations. Tunnel-te1 has a metric value of 2 and tunnel-te2 has a metric value of 1.
- R1 can reach the anycast prefix (10.1.1.7/32) using two paths R1-R2-R4 and R1-R3-R5-R6.
- To reach the anycast prefix, the IGP cost is 20 for the R1-R2-R4 path and it is 30 for the R1-R3-R5-R6 path. Based on the IGP metric, the R1-R2-R4 path is the shortest path.
- With IS-IS, by default, the AA tunnel considers the tunnel metric to forward the traffic. In this topology, tunnel-te2 has a lower metric value when compared to tunnel-te1. As the AA metric is considered and the traffic is forwarded through te-2.
- When this feature is enabled, the IGP cost is first considered for the shortest path like AA with OSPF.

  In this topology, the R1-R2-R4 path is considered as the IGP cost is 20 that lower when compared to tunnel-te2, which is 30. Then the MPLS-TE tunnel on that path is considered, which is tunnel-te1.

Configure IPv6-only autoroute announce over MPLS-TE tunnels

Perform these steps to configure IPv6-only autoroute announce over MPLS-TE tunnels:

Procedure

Step 1 **configure**

Step 2 **interface tunnel-te** *tunnel-id*

Example:

```
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
```

Configures an MPLS-TE tunnel interface.

Step 3 **autoroute announce include-ipv6**

Example:

```
RP/0/RP0/CPU0:router(config-if)# autoroute announce include-ipv6
```

Enables IPv6 autoroute announce.

Step 4 **exclude-ipv4**

Example:

```
RP/0/RP0/CPU0:router(config-if-tunte-aa)# exclude-ipv4
```

Disables IPv4 autoroute announce.

Step 5 **commit**

Step 6 (Optional) **show mpls traffic-eng autoroute**

Example:

```
RP/0/RP0/CPU0:router#show mpls traffic-eng autoroute

  Destination 192.168.0.2 has 1 tunnels in IS-IS ring level 1
    T1 (traffic share 0, nexthop 192.168.0.3,  metric 0)
    (IS-IS ring level-1, IPV6 Unicast)
    Signalled-Name: T1
```

Verifies that the tunnel announces IPv6-only autoroute information.

Configure IGP path selection for anycast prefixes

This procedure provides the steps to configure MPLS RSVP-TE and enable **anycast-prefer-igp-cost** with IS-IS.

Procedure

Step 1 Configure MPLS RSVP-TE.

Example:

```
/* Configure MPLS RSVP-TE */
Router(config)#interface tunnel-te1
```

```
Router(config-if)#ipv4 unnumbered Loopback0
Router(config-if)#autoroute announce
Router(config-if-tunte-aa)#metric 2
Router(config-if-tunte-aa)#exit
Router(config-if)#destination 10.1.1.4
Router(config-if)#path-option 1 dynamic
Router(config-if)#commit

Router(config)#interface tunnel-te2
Router(config-if)#ipv4 unnumbered Loopback0
Router(config-if)#autoroute announce
Router(config-if-tunte-aa)#metric 1
Router(config-if-tunte-aa)#exit
Router(config-if)#destination 10.1.1.6
Router(config-if)#path-option 1 dynamic
Router(config-if)#exit
Router(config)#commit
```

Step 2 Enable **anycast-prefer-igp-cost** with IS-IS.

Example:

```
Router#configure
Router(config)#router isis 100
Router(config-isis)#is-type level-2-only
Router(config-isis)#net 47.2377.50ea.ffff.988a.2d13.00
Router(config-isis)#address-family ipv4 unicast
Router(config-isis-af)#metric-style wide
Router(config-isis-af)#mpls traffic-eng level-2-only
Router(config-isis-af)#mpls traffic-eng router-id Loopback0
Router(config-isis-af)#mpls traffic-eng tunnel anycast-prefer-igp-cost
Router(config-isis-af)#maximum-paths 64
Router(config-isis-af)#commit
```

Step 3 Use the **show running config** to view the running configuration details.

Example:

```
Router# show running-config
interface tunnel-te1
 ipv4 unnumbered Loopback0
 autoroute announce
  metric 2
 !
 destination 10.1.1.4
 path-option 1 dynamic
!
interface tunnel-te2
 ipv4 unnumbered Loopback0
 autoroute announce
  metric 1
 !
 destination 10.1.1.6
 path-option 1 dynamic
!
router isis 100
 is-type level-2-only
 net 47.2377.50ea.ffff.988a.2d13.00
 address-family ipv4 unicast
  metric-style wide
  mpls traffic-eng level-2-only
  mpls traffic-eng router-id Loopback0
  mpls traffic-eng tunnel anycast-prefer-igp-cost
  maximum-paths 64
```

```
 !
!
end
```

Step 4 Execute the **show route** command to verify the configuration.

- When you enable **anycast-prefer-igp-cost**, the traffic is forwarded through tunnel-te1:

```
Router#show route 10.1.1.7/32
Routing entry for 10.1.1.7/32
  Known via "isis 100", distance 115, metric 12, type level-2
  Installed Nov  3 09:48:39.520 for 00:00:05
  Routing Descriptor Blocks
    10.1.1.4, from 10.1.1.4, via tunnel-te1
      Route metric is 20
```

- When you disable **anycast-prefer-igp-cost**, the traffic is forwarded through tunnel-te2:

```
Router#show route 10.1.1.7/32
Routing entry for 10.1.1.7/32
  Known via "isis 100", distance 115, metric 11, type level-2
  Installed Nov  3 09:25:38.162 for 00:18:23
  Routing Descriptor Blocks
    10.1.1.6, from 10.1.1.6, via tunnel-te2
      Route metric is 11
```

Configuring Auto-Tunnel Mesh

The MPLS-TE auto-tunnel mesh (auto-mesh) feature allows you to set up full mesh of TE Point-to-Point (P2P) tunnels automatically with a minimal set of MPLS traffic engineering configurations. You can configure one or more mesh-groups and each mesh-group requires a destination-list (IPv4 prefix-list) listing destinations, which are used as destinations for creating tunnels for that mesh-group.

You can configure MPLS-TE auto-mesh type attribute-sets (templates) and associate them to mesh-groups. Label Switching Routers (LSRs) can create tunnels using the tunnel properties defined in this attribute-set.

Auto-Tunnel mesh configuration minimizes the initial configuration of the network. You can configure tunnel properties template and mesh-groups or destination-lists on TE LSRs that further creates full mesh of TE tunnels between those LSRs. It eliminates the need to reconfigure each existing TE LSR in order to establish a full mesh of TE tunnels whenever a new TE LSR is added in the network.

Configuration Example

This example configures an auto-tunnel mesh group and specifies the attributes for the tunnels in the mesh-group.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel mesh
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh)# tunnel-id min 1000 max 2000
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh)# group 10
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh-group)# attribute-set 10
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh-group)# destination-list dl-65
RP/0/RP0/CPU0:router(config-mpls-te)# attribute-set auto-mesh 10
RP/0/RP0/CPU0:router(config-mpls-te-attribute-set)# autoroute announce
RP/0/RP0/CPU0:router(config-mpls-te-attribute-set)# auto-bw collect-bw-only
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Verify the auto-tunnel mesh configuration using the **show mpls traffic-eng auto-tunnel mesh** command.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng auto-tunnel mesh

Auto-tunnel Mesh Global Configuration:
  Unused removal timeout: 1h 0m 0s
  Configured tunnel number range: 1000-2000

Auto-tunnel Mesh Groups Summary:
  Mesh Groups count: 1
  Mesh Groups Destinations count: 3
  Mesh Groups Tunnels count:
    3 created, 3 up, 0 down, 0 FRR enabled

Mesh Group: 10 (3 Destinations)
  Status: Enabled
  Attribute-set: 10
  Destination-list: dl-65 (Not a prefix-list)
  Recreate timer: Not running
       Destination    Tunnel ID    State  Unused timer
  ----------------  -----------  -------  ------------
       192.168.0.2         1000     up   Not running
       192.168.0.3         1001     up   Not running
       192.168.0.4         1002     up   Not running
  Displayed 3 tunnels, 3 up, 0 down, 0 FRR enabled

Auto-mesh Cumulative Counters:
  Last cleared: Wed Oct  3 12:56:37 2015 (02:39:07 ago)
                      Total
  Created:                3
  Connected:              0
  Removed (unused):       0
  Removed (in use):       0
  Range exceeded:         0
```

Configuring an MPLS Traffic Engineering Interarea Tunneling

The MPLS TE Interarea Tunneling feature allows you to establish MPLS TE tunnels that span multiple Interior Gateway Protocol (IGP) areas and levels. This feature removes the restriction that required the tunnel headend and tailend routers both to be in the same area. The IGP can be either Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF).To configure an inter-area tunnel, you specify on the headend router a loosely routed explicit path for the tunnel label switched path (LSP) that identifies each area border router (ABR) the LSP should traverse using the next-address loose command. The headend router and the ABRs along the specified explicit path expand the loose hops, each computing the path segment to the next ABR or tunnel destination.

Configuration Example

This example configures an IPv4 explicit path with ABR configured as loose address on the headend router.

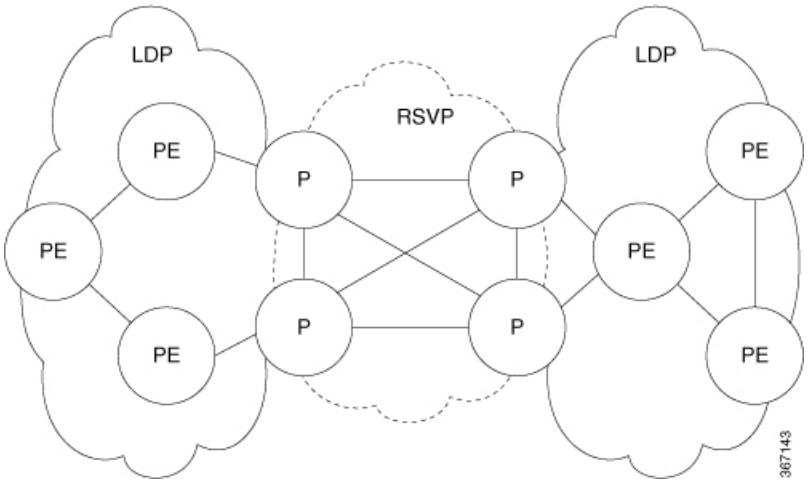
```
Router# configure
Router(config)# explicit-path name interarea1
Router(config-expl-path)# index1 next-address loose ipv4 unicast 172.16.255.129
Router(config-expl-path)# index 2 next-address loose ipv4 unicast 172.16.255.131
Router(config)# interface tunnel-te1
Router(config-if)# ipv4 unnumbered Loopback0
Router(config-if)# destination 172.16.255.2
Router(config-if)# path-option 10 explicit name interarea1
Router(config)# commit
```

Configuring LDP over MPLS-TE

LDP and RSVP-TE are signaling protocols used for establishing LSPs in MPLS networks. While LDP is easy to configure and reilable, it lacks the traffic engineering capabilities of RSVP that helps to avoid traffic congestions. LDP over MPLS-TE feature combines the benefits of both LDP and RSVP. In LDP over MPLS-TE, an LDP signalled label-switched path (LSP) runs through a TE tunnel established using RSVP-TE.

The following diagram explains a use case for LDP over MPLS-TE. In this diagram, LDP is used as the signalling protocol between provider edge (PE) router and provider (P) router. RSVP-TE is used as the signalling protocol between the P routers to establish an LSP. LDP is tunneled over the RSVP-TE LSP.

Figure 14: LDP over MPLS-TE



Restrictions and Guidelines for LDP over MPLS-TE

The following restrictions and guidelines apply for this feature in Cisco IOS-XR release 6.3.2:

- MPLS services over LDP over MPLS-TE are supported when BGP neighbours are on the head or tail node of the TE tunnel.
- MPLS services over LDP over MPLS-TE are supported when the TE headend router is acting as transit point for that service.
- If MPLS services are originating from the TE headend, but the TE tunnel is ending before the BGP peer, LDP over MPLS-TE feature is not supported.
- If LDP optimization is enabled using the **hw-module fib mpls ldp lsr-optimized** command, the following restrictions apply:
  - EVPN is not supported.
  - For any prefix or label all outgoing paths has to be LDP enabled.
- Do not use the **hw-module fib mpls ldp lsr-optimized** command on a Provider Edge (PE) router because already configured features such as EVPN, MPLS-VPN, and L2VPN might not work properly.

Configuration Example:

This example shows how to configure an MPLS-TE tunnel from provider router P1 to P2 and then enbale LDP over MPLS-TE. In this example, the destination of the tunnel from P1 is configured as the loop back for P2.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# autoroute announce
RP/0/RP0/CPU0:router(config-if)# destination 4.4.4.4
RP/0/RP0/CPU0:router(config-if)# path-option 1 dynamic
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:Router(config)#  mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)#  router-id 192.168.1.1
RP/0/RP0/CPU0:Router(config-ldp)#  interface TenGigE 0/0/0/0
RP/0/RP0/CPU0:Router(config-ldp-if)# interface tunnel-te 1
RP/0/RP0/CPU0:Router(config-ldp-if)# exit
```

Configuring MPLS-TE Path Protection

Path protection provides an end-to-end failure recovery mechanism for MPLS-TE tunnels. A secondary Label Switched Path (LSP) is established, in advance, to provide failure protection for the protected LSP that is carrying a tunnel's TE traffic. When there is a failure on the protected LSP, the source router immediately enables the secondary LSP to temporarily carry the tunnel's traffic. Failover is triggered by a RSVP error message sent to the LSP head end. Once the head end received this error message, it switches over to the secondary tunnel. 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 within a single area (OSPF or IS-IS), external BGP [eBGP], and static routes. Both the explicit and dynamic path-options are supported for the MPLS-TE path protection feature. You should make sure that the same attributes or bandwidth requirements are configured on the protected option.

Before You Begin

The following prerequisites are required for enabling path protection.

- You should ensure that your network supports MPLS-TE, Cisco Express Forwarding, and Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF).
- You should configure MPLS-TE on the routers.

Configuration Example

This example configures how to configure path protection for a mpls-te tunnel. The primary path-option should be present to configure path protection. In this configuration, R1 is the headend router and R3 is the tailend router for the tunnel while R2 and R4 are mid-point routers. In this example, 6 explicit paths and 1 dynamic path is created for path protection. You can have upto 8 path protection options for a primary path.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 0
RP/0/RP0/CPU0:router(config-if)# destination 192.168.3.3
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# autoroute announce
RP/0/RP0/CPU0:router(config-if)# path-protection
RP/0/RP0/CPU0:router(config-if)# path-option 1 explicit name r1-r2-r3-00 protected-by 2
RP/0/RP0/CPU0:router(config-if)# path-option 2 explicit name r1-r2-r3-01 protected-by 3
RP/0/RP0/CPU0:router(config-if)# path-option 3 explicit name r1-r4-r3-01 protected-by 4
RP/0/RP0/CPU0:router(config-if)# path-option 4 explicit name r1-r3-00 protected-by 5
```

```
RP/0/RP0/CPU0:router(config-if)# path-option 5 explicit name r1-r2-r4-r3-00 protected-by 6
RP/0/RP0/CPU0:router(config-if)# path-option 6 explicit name r1-r4-r2-r3-00 protected-by 7
RP/0/RP0/CPU0:router(config-if)# path-option 7 dynamic
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Use the **show mpls traffic-eng tunnels** command to verify the MPLS-TE path protection configuration.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels 0
Fri Oct 13 16:24:39.379 UTC
Name: tunnel-te0  Destination: 192.168.92.125  Ifhandle:0x8007d34
  Signalled-Name: router
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected
    path option 1,  type explicit r1-r2-r3-00 (Basis for Setup, path weight 2)
      Protected-by PO index: 2
    path option 2,  type explicit r1-r2-r3-01 (Basis for Standby, path weight 2)
      Protected-by PO index: 3
    path option 3,  type explicit r1-r4-r3-01
      Protected-by PO index: 4
    path option 4,  type explicit r1-r3-00
      Protected-by PO index: 5
    path option 5,  type explicit r1-r2-r4-r3-00
      Protected-by PO index: 6
    path option 6,  type explicit r1-r4-r2-r3-00
      Protected-by PO index: 7
    path option 7,  type dynamic
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Fri Oct 13 15:05:28 2017 (01:19:11 ago)
  Config Parameters:
    Bandwidth:        0 kbps (CT0) Priority:  7  7 Affinity: 0x0/0xffff
    Metric Type: TE (global)
    Path Selection:
      Tiebreaker: Min-fill (default)
    Hop-limit: disabled
    Cost-limit: disabled
    Delay-limit: disabled
    Path-invalidation timeout: 10000 msec (default), Action: Tear (default)
    AutoRoute:  enabled  LockDown: disabled   Policy class: not set
    Forward class: 0 (not enabled)
    Forwarding-Adjacency: disabled
    Autoroute Destinations: 0
    Loadshare:          0 equal loadshares
    Auto-bw: disabled
    Fast Reroute: Disabled, Protection Desired: None
    Path Protection: Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    Soft Preemption: Disabled
  History:
    Tunnel has been up for: 01:14:13 (since Fri Oct 13 15:10:26 UTC 2017)
    Current LSP:
      Uptime: 01:14:13 (since Fri Oct 13 15:10:26 UTC 2017)
    Reopt. LSP:
      Last Failure:
        LSP not signalled, identical to the [CURRENT] LSP
        Date/Time: Fri Oct 13 15:08:41 UTC 2017 [01:15:58 ago]
    Standby Reopt LSP:
      Last Failure:
        LSP not signalled, identical to the [STANDBY] LSP
        Date/Time: Fri Oct 13 15:08:41 UTC 2017 [01:15:58 ago]
```

```
        First Destination Failed: 192.3.3.3
      Prior LSP:
        ID: 8 Path Option: 1
        Removal Trigger: path protection switchover
      Standby LSP:
        Uptime: 01:13:56 (since Fri Oct 13 15:10:43 UTC 2017)
    Path info (OSPF 1 area 0):
    Node hop count: 2
    Hop0: 192.168.1.2
    Hop1: 192.168.3.1
    Hop2: 192.168.3.2
    Hop3: 192.168.3.3
    Standby LSP Path info (OSPF 1 area 0), Oper State: Up :
    Node hop count: 2
    Hop0: 192.168.2.2
    Hop1: 192.168.3.1
    Hop2: 192.168.3.2
    Hop3: 192.168.3.3
  Displayed 1 (of 4001) heads, 0 (of 0) midpoints, 0 (of 0) tails
  Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

Configuring Auto-bandwidth Bundle TE++

Table 20: Feature History Table

Feature Name	Release Information	Feature Description
Automatic Bandwidth Bundle TE++ for Numbered Tunnels	Release 7.10.1	Introduced in this release on: NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5700 line cards [Mode: Compatibility; Native]) We have optimized network performance and enabled efficient utilization of resources for numbered tunnels based on real-time traffic by automatically adding or removing tunnels between two endpoints. This is made possible because this release introduces support for auto-bandwidth TE++ for numbered tunnels, expanding upon the previous support for only named tunnels, letting you define explicit paths and allocate the bandwidth to each tunnel. The feature introduces these changes: • **CLI:** The **auto-capacity** keyword is added to the **interface tunnel-te** command. • **YANG Data Model:** New XPaths for `Cisco-IOS-XR-mpls-te-cfg.yang` (see GitHub, YANG Data Models Navigator)

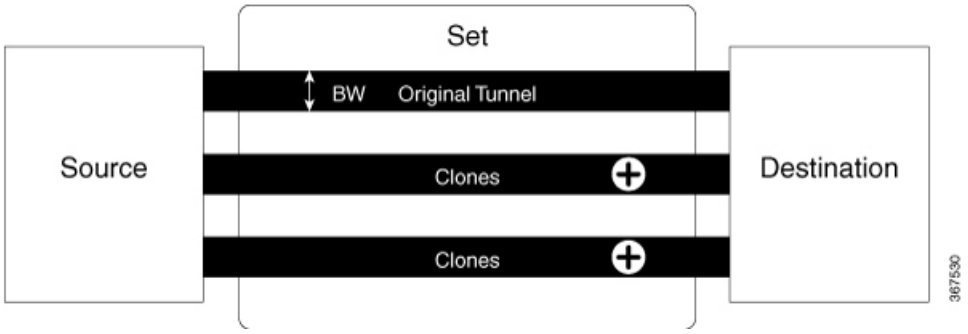
MPLS-TE tunnels are used to set up labeled connectivity and to provide dynamic bandwidth capacity between endpoints. The auto-bandwidth feature addresses the dynamic bandwidth capacity demands by dynamically resizing the MPLS-TE tunnels based on the measured traffic loads. However, many customers require multiple auto-bandwidth tunnels between endpoints for load balancing and redundancy. When the aggregate bandwidth demand increases between two endpoints, you can either configure auto-bandwidth feature to resize the tunnels or create new tunnels and load balance the overall demand over all the tunnels between two endpoints. Similarly, when the aggregate bandwidth demand decreases between two endpoints you can either configure the auto-bandwidth feature to decrease the sizes of the tunnel or delete the new tunnels and load balance the traffic over the remaining tunnels between the endpoints. The autobandwidth bundle TE++ feature is an

extension of the auto-bandwidth feature and allows you to automatically increase or decrease the number of MPLS-TE tunnels to a destination based on real time traffic needs.

Tunnels that are automatically created as a response to the increasing bandwidth demands are called clones. The cloned tunnels inherit properties of the main configured tunnel. However, user configured load interval cannot be inherited. The original tunnel and its clones are collectively called a set. You can specify an upper limit and lower limit on the number of clones that can be created for the original tunnel.

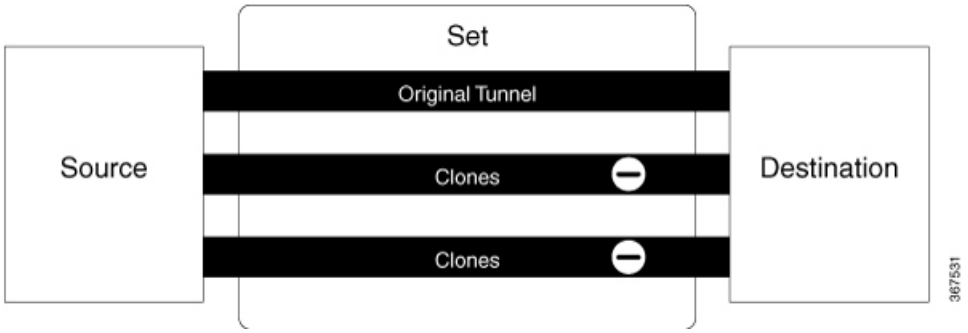
Splitting is the process of cloning a new tunnel when there is a demand for bandwidth increase. When the size of any of the tunnels in the set crosses a configured split bandwidth, then splitting is initiated and clone tunnels are created.

The following figure explains creating clone tunnels when the split bandwidth is exceeded.



Merging is the process of removing a clone tunnel when the bandwidth demand decreases. If the bandwidth goes below the configured merge bandwidth in any one of the tunnels in the set, clone tunnels are removed.

The following figure explains removing clone tunnels to merge with the original tunnel when the bandwidth falls below the merge bandwidth.



There are multiple ways to equally load-share the aggregate bandwidth demand among the tunnels in the set. This means that an algorithm is needed to choose the pair which satisfies the aggregate bandwidth requirements. You can configure a nominal bandwidth to guide the algorithm to determine the average bandwidths of the tunnels. If nominal bandwidth is not configured, TE uses the average of split and merge bandwidth as nominal bandwidth.

Restrictions and Usage Guidelines

The following usage guidelines apply for the auto-bandwidth bundle TE++ feature.

- This feature is only supported for the named tunnels and not supported on tunnel-te interfaces.
- The range for the lower limit on the number of clones is 0 to 63 and the default value for the lower limit on the number of clones is 0.

- The range for the upper limit on the number of clones is 1 to 63 and the default value for the upper limit on the number of clones is 63.

Configuration Example

This example shows how to configure the autobandwidth bundle TE++ feature for a named MPLS-TE traffic tunnel. You should configure the following values for this feature to work:

- min-clones: Specifies the minimum number of clone tunnels that the original tunnel can create.
- max-clones: Specifies the maximum number of clone tunnels that the original tunnel can create.
- nominal-bandwidth: Specifies the average bandwidth for computing the number of tunnels to satisfy the overall demand.
- split-bandwidth: Specifies the bandwidth value for splitting the original tunnel. If the tunnel bandwidth exceeds the configured split bandwidth, clone tunnels are created.
- merge-bandwidth: Specifies the bandwidth for merging clones with the original tunnel. If the bandwidth goes below the configured merge bandwidth, clone tunnels are removed.

In this example, the lower limit on the number of clones is configured as two and the upper limit on the number of clones is configured as four. The bandwidth size for splitting and merging is configured as 200 and 100 kbps.

```
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# named-tunnels
RP/0/RP0/CPU0:router(config-te-named-tunnels)# tunnel-te xyz
RP/0/RP0/CPU0:router(config-te-tun-name)# auto-bw
RP/0/RP0/CPU0:router(config-mpls-te-tun-autobw)# auto-capacity
RP/0/RP0/CPU0:router(config-te-tun-autocapacity)# min-clones 2
RP/0/RP0/CPU0:router(config-te-tun-autocapacity)# max-clones 4
RP/0/RP0/CPU0:router(config-te-tun-autocapacity)# nominal-bandwidth 150
RP/0/RP0/CPU0:router(config-te-tun-autocapacity)# split-bandwidth 200
RP/0/RP0/CPU0:router(config-te-tun-autocapacity)# merge-bandwidth 100
```

Configure Autoroute Tunnel as Designated Path

Table 21: Feature History Table

Feature Name	Release Information	Feature Description
Configure Autoroute Tunnel as Designated Path	Release 7.6.2	We now provide you the flexibility to simplify the path selection for a traffic class and split traffic among multiple TE tunnels. To split traffic, you can specify an autoroute tunnel to forward traffic to a particular tunnel destination address without considering the IS-IS metric for traffic path selection. IS-IS metric provides the shortest IGP path to a destination based only on link costs along the path. However, you may want to specify a tunnel interface to carry traffic regardless of IGP cost to meet your specific organizational requirements. Earlier, MPLS-TE considered either the Forwarding Adjacency (FA) or Autoroute (AA) tunnel for forwarding traffic based only on the IS-IS metric. The feature introduces the **mpls traffic-eng tunnel restricted** command.

MPLS-TE builds a unidirectional tunnel from a source to a destination using label switched path (LSP) to forward traffic.

To forward the traffic through MPLS tunneling, you can use autoroute, forwarding adjacency, or static routing:

- Autoroute (AA) functionality allows to insert the MPLS TE tunnel in the Shortest Path First (SPF) tree for the tunnel to transport all the traffic from the headend to all destinations behind the tail-end. AA is only known to the tunnel headend router.
- Forwarding Adjacency (FA) allows the MPLS-TE tunnel to be advertised as a link in an IGP network with the cost of the link associated with it. Routers outside of the TE domain can see the TE tunnel and use it to compute the shortest path for routing traffic throughout the network.
- Static routing allows you to inject static IP traffic into a tunnel as the output interface for the routing decision.

Prior to this release, by default, MPLS-TE considers FA or AA tunnels to forward traffic based on the IS-IS metric. The lower metric is always used to forward traffic. There was no mechanism to forward traffic to a specific tunnel interface.

For certain prefixes to achieve many benefits such as security and service-level agreements, there might be a need to forward traffic to a specific tunnel interface that has a matching destination address.

With this feature, you can exclusively use AA tunnels to forward traffic to their tunnel destination address irrespective of IS-IS metric. Traffic steering is performed based on the prefixes and not metrics. Traffic to other prefixes defaults to the forwarding-adjacency (FA) tunnels.

To enable this feature, use the **mpls traffic-eng tunnel restricted** command.

Also, you may require more than one AA tunnel to a particular remote PE and use ECMP to forward traffic across AA tunnels. You can configure a loopback interface with one primary address and multiple secondary addresses on the remote PE, using one IP for the FA tunnel destination, and others for the AA tunnels destinations. Multiple IP addresses are advertised in the MPLS TE domain using the typed length value (TLV) 132 in IS-IS. A TLV-encoded data stream contains code related to the record type, the record length of the value, and value. TLV 132 represents the IP addresses of the transmitting interface.

Feature Behavior

When MPLS-TE tunnel restricted is configured, the following is the behavior:

- A complete set of candidate paths is available for selection on a per-prefix basis during RIB update as the first hop computation includes all the AA tunnels terminating on a node up to a limit of 64 and the lowest cost forwarding-adjacency or native paths terminating on the node or inherited from the parent nodes in the first hops set for the node.
- During per-prefix computation, AA tunnel first hops are used for traffic sent to their tunnel destination address even if FA tunnel or native first hops have a better metric. AA tunnel first-hops are not used for any other prefixes.
- ECMP is used when multiple AA tunnel first hops have the same destination address and metric.
- During per-prefix computation, AA tunnel first hops are used for traffic sent to their tunnel destination address, and for all other destinations on the tunnel tail node or behind it, even if a native path has a better metric.

  Adding **mpls traffic-eng tunnel preferred** configuration has no effect when the tunnel restricted is already configured.
- If there's no AA tunnel or if the tunnel is down, then native paths are used for all other destinations on the tunnel tail node or behind it.

  The route metric for a prefix reflects the chosen first-hop, not necessarily the lowest cost SPF distance to the node.

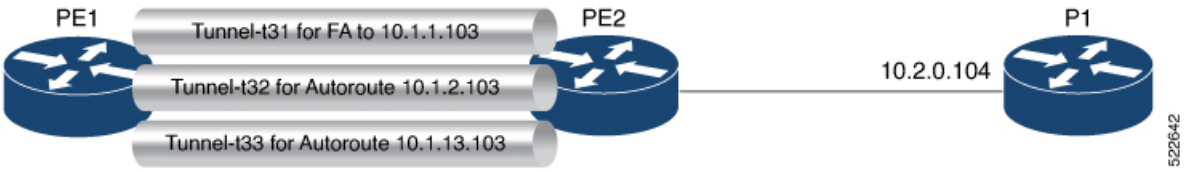
Restrictions for Configure Autoroute Tunnel as Designated Path

- The total number of interface addresses to the number that can be contained in 255 bytes is 63 for IPv4 and 15 for IPv6.
- When this feature is enabled, a maximum of 64 tunnels can terminate on the tail node.

Configure Autoroute Tunnel as Designated Path

Let's understand how to configure the feature using the following topology:

Figure 15: Topology



Consider the topology where PE1 has three MPLS tunnels connecting to PE2.

- Tunnel-t31: Forwarding adjacency (FA) is configured to the primary address of Loopback 0 on PE2 (10.1.1.103).
- Tunnel- t32: Autoroute announce (AA) is configured to a secondary address of Loopback 0 on PE2 (10.1.2.103).
- Tunnel-t33: Autoroute announce (AA) is configured to a secondary address of Loopback 0 on PE2 (10.1.3.103).

This feature is not enabled by default. When this feature is not enabled, traffic is load balanced over all AA tunnels towards the same remote PE provided the tunnel metric is the same:

```
Router# show routes
i L2 10.1.1.103/32 [115/40] via 10.1.2.103, 00:00:30, tunnel-t32
                   [115/40] via 10.1.3.103, 00:00:30, tunnel-t33
i L2 10.1.2.103/32 [115/40] via 10.1.2.103, 00:00:30, tunnel-t32
                   [115/40] via 10.1.3.103, 00:00:30, tunnel-t33
i L2 10.1.3.103/32 [115/40] via 10.1.2.103, 00:00:30, tunnel-t32
                   [115/40] via 10.1.3.103, 00:00:30, tunnel-t33
i L2 10.2.0.103/32 [115/40] via 10.1.2.103, 00:00:30, tunnel-t32
                   [115/40] via 10.1.3.103, 00:00:30, tunnel-t33
 10.2.0.104/32 [115/50] via 10.1.2.103, 00:00:30, tunnel-t32
                   [115/50] via 10.1.3.103, 00:00:30, tunnel-t33
```

Configuration Example

You can configure the feature using the **mpls traffic-eng tunnel restricted** command.

```
RP/0/RSP0/CPU0:ios# configure
RP/0/RSP0/CPU0:ios(config)# router isis 1
RP/0/RSP0/CPU0:ios(config-isis)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-af# mpls traffic-eng tunnel restricted
```

Running Configuration

The following example shows the AA tunnel metric running configuration:

```
router isis 1
 address-family ipv4 unicast
  mpls traffic-eng tunnel restricted
 !
!
end
```

Verification

When you enable the feature, traffic towards a particular prefix is sent only over the tunnel that has that IP address as destination.

```
Router# show route
i L2 10.1.1.103/32 [115/40] via 10.1.1.103, 00:00:04, tunnel-t31
i L2 10.1.2.103/32 [115/40] via 10.1.2.103, 00:00:04, tunnel-t32
i L2 10.1.3.103/32 [115/40] via 10.1.3.103, 00:00:04, tunnel-t33
i L2 10.2.0.103/32 [115/40] via 10.1.1.103, 00:00:04, tunnel-t31
i L2 10.2.0.104/32 [115/50] via 10.1.1.103, 00:00:04, tunnel-t31
```

When multiple restricted AA tunnels are created towards the same destination IP address, router load balances traffic across all those tunnels:

```
Router# show route
i L2 10.1.1.103/32 [115/40] via 10.1.1.101, 00:00:08, GigabitEthernet0/0/0/2
                   [115/40] via 10.1.3.101, 00:00:08, GigabitEthernet0/0/0/3
i L2 10.1.2.103/32 [115/40] via 10.1.2.103, 00:00:08, tunnel-t32
                   [115/40] via 10.1.2.103, 00:00:30, tunnel-t34
i L2 10.1.3.103/32 [115/40] via 10.1.3.103, 00:00:08, tunnel-t33
i L2 10.2.0.103/32 [115/40] via 10.1.1.101, 00:00:08, GigabitEthernet0/0/0/2
                   [115/40] via 10.1.3.101, 00:00:08, GigabitEthernet0/0/0/3
i L2 10.2.0.104/32 [115/50] via 10.1.1.101, 00:00:08, GigabitEthernet0/0/0/2
                   [115/50] via 10.1.3.101, 00:00:08, GigabitEthernet0/0/0/3
```

Configure IPv6 routing over IPv4 MPLS-TE tunnels

Perform these steps to configure IPv6 routing over IPv4 MPLS-TE tunnels:

Procedure

Step 1 **configure**

Step 2 **interface tunnel-te** *tunnel-id*

Example:

```
RP/0/RP0/CPU0:router# interface tunnel-te 1
```

Configures an MPLS-TE tunnel interface.

Step 3 **ipv4 unnumbered** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-if)#ipv4 unnumbered Loopback 0
```

Assigns a source address so that forwarding can be performed on the new tunnel. Loopback is the commonly-used interface type.

Step 4 **ipv6 enable**

Example:

```
RP/0/RP0/CPU0:router(config-if)#ipv6 enable
```

Enables IPv6 on interface.

Step 5 **signalled-bandwidth** *bandwidth*

Example:

```
RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 10
```

Sets the tunnel bandwidth requirement to be signalled in Kbps.

Step 6 **destination** *ip-address*

Example:

```
RP/0/RP0/CPU0:router(config-if)#destination 192.168.0.1
```

Specifies the IPv4 destination address.

Step 7 Use one of these options:

- **autoroute announce include-ipv6**
- **forwarding-adjacency include-ipv6**

Example:

```
RP/0/RP0/CPU0:router(config-if)#autoroute announce include-ipv6
```

Or

```
RP/0/RP0/CPU0:router(config-if)#forwarding-adjacency include-ipv6
```

Announces the tunnel as an IPv6 autoroute, or as an IPv6 forwarding adjacency.

Step 8 **path-option** *preference-priority* **dynamic**

Example:

```
RP/0/RP0/CPU0:router(config-if)#path-option 1 dynamic
```

Sets the path option to dynamic and assigns the path ID.

Step 9 **commit**

Step 10 (Optional) **show mpls traffic-eng autoroute**

Example:

```
RP/0/RP0/CPU0:router#show mpls traffic-eng autoroute

  Destination 192.168.0.2 has 1 tunnels in IS-IS ring level 1
    tunnel-te1 (traffic share 0, nexthop 192.168.0.2)
               (IPv4 unicast)
               (IPv6 unicast)
```

Verifies that the tunnel announces IPv6 autoroute information.

Step 11 (Optional) **show mpls traffic-eng forwarding-adjacency**

Example:

```
RP/0/RP0/CPU0:router#show mpls traffic-eng forwarding-adjacency

 destination 192.168.0.1 has 1 tunnels

              tunnel-te10    (traffic share 0, next-hop 192.168.0.1)
                             (Adjacency Announced: yes, holdtime 0)
                             (IS-IS 100, IPv4 unicast)
                             (IS-IS 100, IPv6 unicast)
```

Verifies that the tunnel announces IPv6 forwarding adjacency information.

Configure IPv6 routing over IPv6 MPLS-TE tunnels

Perform these steps to configure IPv6 routing over IPv6 MPLS-TE tunnels:

Procedure

Step 1 **configure**

Step 2 **interface tunnel-te** *tunnel-id*

Example:

```
RP/0/RP0/CPU0:router# interface tunnel-te 1
```

Configures an MPLS-TE tunnel interface.

Step 3 **ipv4 unnumbered** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-if)#ipv4 unnumbered Loopback 0
```

Assigns a source address so that forwarding can be performed on the new tunnel. Loopback is the commonly-used interface type.

Step 4 **ipv6 enable**

Example:

```
RP/0/RP0/CPU0:router(config-if)#ipv6 enable
```

Enables IPv6 on interface.

Step 5 **signalled-bandwidth** *bandwidth*

Example:

```
RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 10
```

Sets the tunnel bandwidth requirement to be signalled in Kbps.

Step 6 Use one of these options:

- **autoroute announce include-ipv6**
- **forwarding-adjacency include-ipv6**

Example:

```
RP/0/RP0/CPU0:router(config-if)#autoroute announce include-ipv6
```

Or

```
RP/0/RP0/CPU0:router(config-if)#forwarding-adjacency include-ipv6
```

Announces the tunnel as an IPv6 autoroute, or as an IPv6 forwarding adjacency.

Step 7 **destination ipv6** *ip-address*

Example:

```
RP/0/RP0/CPU0:router(config-if)#destination ipv6 2000::1
```

Specifies the IPv6 destination address.

Step 8 **path-option** *preference-priority* **dynamic**

Example:

```
RP/0/RP0/CPU0:router(config-if)#path-option 1 dynamic
```

Sets the path option to dynamic and assigns the path ID.

Step 9 **commit**

Step 10 (Optional) **show mpls traffic-eng autoroute**

Example:

```
RP/0/RP0/CPU0:router#show mpls traffic-eng autoroute

  Destination 192.168.0.2 has 1 tunnels in IS-IS ring level 1
    tunnel-te1 (traffic share 0, nexthop 192.168.0.2)
               (IPv4 unicast)
               (IPv6 unicast)
```

Verifies that the tunnel announces IPv6 autoroute information.

Step 11 (Optional) **show mpls traffic-eng forwarding-adjacency**

Example:

```
RP/0/RP0/CPU0:router#show mpls traffic-eng forwarding-adjacency

 destination 192.168.0.1 has 1 tunnels

              tunnel-te10    (traffic share 0, next-hop 192.168.0.1)
                             (Adjacency Announced: yes, holdtime 0)
                             (IS-IS 100, IPv4 unicast)
                             (IS-IS 100, IPv6 unicast)
```

Verifies that the tunnel announces IPv6 forwarding adjacency information.

Regular expressions in MPLS-TE tunnel names

In large scale MPLS-TE tunnel deployments, regular expressions (regex) can provide a scalable way to filter tunnels for visibility, run actions on group of tunnels, and avoid issuing commands one tunnel at a time.

Table 22: Feature History Table

Feature Name	Release Information	Feature Description
Regular expressions in MPLS-TE tunnel names	Release 26.2.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]) This feature enhances operational scalability and usability by allowing regex-based tunnel name filtering for MPLS-TE show and execution commands, without changing configuration models, architecture, or security posture. This feature supports regular expression in all MPLS-TE show and execution commands using tunnel names.

Starting from Cisco IOS XR Release 26.2.1, you can use regular expressions in execution and show commands with MPLS-TE tunnel names to filter multiple tunnels at a time.

Regular expressions enable operators to quickly filter show command output for MPLS-TE tunnel names, allowing them to

- view all tunnels for a specific region, customer, or service
- reduce noise in large outputs, and
- troubleshoot issues more quickly during outages or convergence events

By using regular expressions, operators can identify all related MPLS-TE tunnels at once, streamlining operational tasks across large-scale deployments.

Benefits of using regular expressions in MPLS-TE tunnel names

- Manage all related tunnels at once in large scale deployments.
- Apply actions to multiple tunnels in a single command, such as reoptimizing paths, rerouting tunnels, and clearing auto-bandwidth state.
- Leverage existing naming conventions without changing tunnel configuration.
- Allow fine-grained selection preventing incorrect matches in large MPLS-TE networks.

Limitations of using regex in MPLS-TE tunnels

The gNMI, Netconf, and programmability interfaces do not support regular expressions.

Usage guidelines of regex in MPLS-TE tunnel names

Follow these guidelines to use regex with MPLS-TE tunnel names:

- Tunnel names support the dot character ('.'). However, using `.` as a regex in tunnel names does not match any single character. To filter tunnel names in this way, use regex patterns such as `b[0-9a-zA-Z_-]r` or `b(.)r`.

  `.*` and `.+` are exceptions that are considered as regular expressions matching any character N times.

- When your regex includes '(', ')', or '|', enclose the pattern in double quotes. Without double quotes, it will be reported as an unsupported character.

```
Router# mpls traffic-eng reoptimize name "(FROM-nyc.*)"
Router# mpls traffic-eng reoptimize name "(FROM-nyc|To-LA)"
```

- Regex patterns containing backslash (\) do not work because the IOS XR parser removes all backslash characters from the string before processing.
  - Use `[0-9]` instead of `\d`
  - Use `[A-Za-z0-9_]` instead of `\w`
  - Using a dot (.) matches literal `'.'`, and there is no need to use `\.`

    Example using `\w`:

```
Router# show mpls traffic-eng tunnels name fo.\w brief
TUNNEL NAME         DESTINATION      STATUS  STATE
       fo.w         192.168.0.3          up     up
```

    Example using `.`(dot):

```
Router# show mpls traffic-eng tunnels name fo.o brief

TUNNEL NAME         DESTINATION      STATUS  STATE
       fo.o         192.168.0.3          up     up

Router# show mpls traffic-eng tunnels name fo[.f]o brief

TUNNEL NAME         DESTINATION      STATUS  STATE
       fo.o         192.168.0.3          up     up
       fofo         192.168.0.3          up     up
```

- For backward compatibility, all patterns are automatically wrapped as `^pattern$`.
- To match substrings, explicitly include wildcards such as `.*pattern.*`.

  Example:

```
/* Matches only the tunnel name "foo". */
Router# show mpls traffic-eng tunnels name foo

/* Matches all tunnels containing "foo" */
Router# show mpls traffic-eng tunnels name .*foo.*
```

- The question mark ('?') is interpreted by the CLI for command help or auto-completion. Hence, avoid using `?` in regex.

Table 23: Regex summary

Typical regex	Parser behavior	Usage
`'.'`	Literal only	Use `'.'` or character class such as `[a-zA-Z0-9_-.]`
`'(', ')', '|'`	Not allowed	Not allowed unless inside double quotes
`\`	Stripped	Use explicit character classes
`?`	CLI help character	Do not use
`^$`	Auto-applied	Explicit use has no impact
Substring match	Disabled by default	Use `.*pattern.*`

By following these guidelines, you ensure your regex patterns for MPLS-TE tunnel names are parsed and executed as intended in show and execution commands.

Regular expressions in MPLS-TE tunnel name filtering commands

You can filter and manage MPLS-TE tunnels efficiently by specifying name patterns with regular expressions in CLI commands. Regular expressions allow you to match tunnels based on their naming patterns for more effective tunnel management.

Supported show commands and execution commands

Execution commands	Show commands
mpls traffic-eng reoptimize name <*tunn-name with regex*>	**show mpls traffic-eng tunnels name** <*tunn-name with regex*> **[affinity | auto-bw | brief | detail | fsm | name-lookup | signame | source | standby | tabular]**
mpls traffic-eng tunnels resetup name <*tunn-name with regex*>	**show mpls traffic-eng event-history tunnels name** <*tunn-name with regex*>
mpls traffic-eng tunnels reroute name <*tunn-name with regex*>	**show mpls traffic-eng autoroute name** <*tunn-name with regex*> **[ip | standby]**
mpls traffic-eng tunnels reroute name <*tunn-name with regex*>	**show mpls traffic-eng autoroute -destination name** <*tunn-name with regex*> **[ip | standby]**
clear mpls traffic-eng auto-bw name <*tunn-name with regex*>	**show mpls traffic-eng tunnels protection [frr] name** <*tunn-name with regex*> **[name-lookup | standby]**
	show mpls traffic-eng tunnels name <*tunn-name with regex*> **role [all | head | middle | tail]**

Some examples include:

- Execution commands:

```
Router# mpls traffic-eng reoptimize name FROM-nyc.*
Router# mpls traffic-eng tunnels reroute name .*TO-LA
```

- Show commands:

```
Router# show mpls traffic-eng tunnels name FROM-nyc.*
Router# show mpls traffic-eng tunnels name .*TO-LA
Router# show mpls traffic-eng tunnels name .*CUSTOMER-A.*
```

These examples demonstrate how to use regular expressions, such as '.*', to match tunnels that begin with , end with, or contain specific name patterns.

MPLS-TE Features - Details

MPLS TE Fast Reroute Link and Node Protection

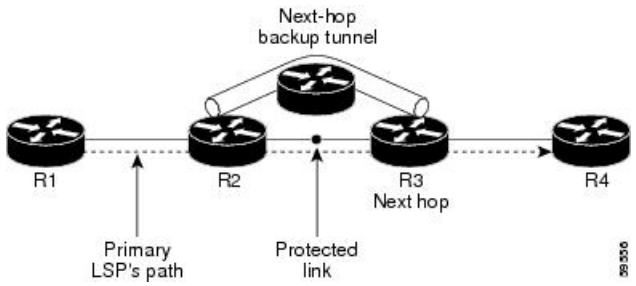
Fast Reroute (FRR) is a mechanism for protecting MPLS TE LSPs from link and node failures by locally repairing the LSPs at the point of failure, allowing data to continue to flow on them while their headend routers try to establish new end-to-end LSPs to replace them. FRR locally repairs the protected LSPs by rerouting them over backup tunnels that bypass failed links or node.

Note If FRR is greater than 50ms, it might lead to a loss of traffic.

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.

The following figure illustrates link protection.

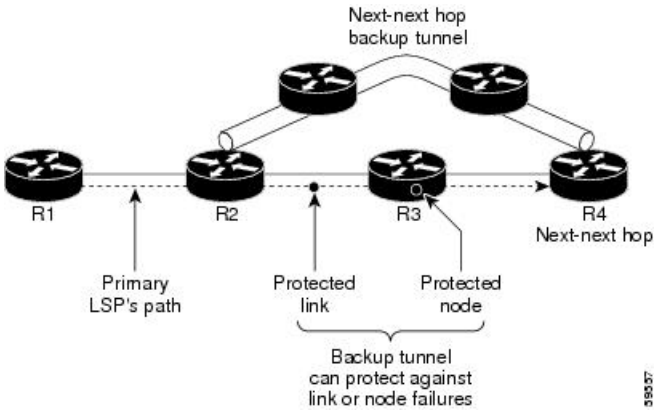
Figure 16: Link Protection



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. NNHOP backup tunnels also provide protection from link failures, because they bypass the failed link and the node.

The following figure illustrates node protection.

Figure 17: Node Protection



Differentiated Services Traffic Engineering

MPLS Differentiated Services Aware Traffic Engineering (DS-TE) is an extension of the regular MPLS-TE feature. Regular traffic engineering does not provide bandwidth guarantees to different traffic classes. A single bandwidth constraint is used in regular TE that is shared by all traffic. To support various classes of service (CoS), you can configure multiple bandwidth constraints. These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint.

Cisco IOS XR software supports two DS-TE modes: pre-standard and IETF. The pre-standard DS-TE mode uses the Cisco proprietary mechanisms for RSVP signaling and IGP advertisements. This DS-TE mode does not interoperate with third-party vendor equipment. Pre-standard DS-TE is enabled only after configuring the sub-pool bandwidth values on MPLS-enabled interfaces. Pre-standard DS-TE mode supports a single bandwidth constraint model a Russian Doll Model (RDM) with two bandwidth pools: global-pool and sub-pool. TE class map is not used with Pre-standard DS-TE mode.

IETF DS-TE mode uses IETF-defined extensions for RSVP and IGP. This mode inter-operates with third-party vendor equipment. IETF mode supports multiple bandwidth constraint models, including RDM and Maximum Allocation Bandwidth Constraint Model (MAM), both with two bandwidth pools. In an IETF DS-TE network, identical bandwidth constraint models must be configured on all nodes. TE class map is used with IETF DS-TE mode and must be configured the same way on all nodes in the network.

The MAM constraint model has the following characteristics:

- Easy to use and intuitive.
- Isolation across class types.
- Simultaneously achieves isolation, bandwidth efficiency, and protection against QoS degradation.

The RDM constraint model has these characteristics:

- Allows greater sharing of bandwidth among different class types.
- Ensures bandwidth efficiency simultaneously and protection against QoS degradation of all class types.
- Specifies that it is used with preemption to simultaneously achieve isolation across class-types such that each class-type is guaranteed its share of bandwidth, bandwidth efficiency, and protection against QoS degradation of all class types.

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 routers regardless of their location in the network. The routers can be located multiple hops from each other.

As a result, a TE tunnel is advertised as a link in an IGP network with the tunnel's cost associated with it. Routers 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. Routers can then use these advertisements in their IGPs to compute the SPF even if they are not the headend of any TE tunnels.

Automatic Bandwidth

Automatic bandwidth allows you to dynamically adjust bandwidth reservation based on measured traffic. MPLS-TE automatic bandwidth is configured on individual Label Switched Paths (LSPs) at every headend router. MPLS-TE automatic bandwidth monitors the traffic rate on a tunnel interface and resizes the bandwidth on the tunnel interface to align it closely with the traffic in the tunnel.

MPLS-TE automatic bandwidth can perform these functions:

- Monitors periodic polling of the tunnel output rate
- Resizes the tunnel bandwidth by adjusting the highest rate observed during a given period.

For every traffic-engineered tunnel that is configured for an automatic bandwidth, the average output rate is sampled, based on various configurable parameters. Then, the tunnel bandwidth is readjusted automatically based on either the largest average output rate that was noticed during a certain interval, or a configured maximum bandwidth value.

While re-optimizing the LSP with the new bandwidth, a new path request is generated. If the new bandwidth is not available, the last good LSP remains used. This way, the network experiences no traffic interruptions. If minimum or maximum bandwidth values are configured for a tunnel, the bandwidth, which the automatic bandwidth signals, stays within these values.

The output rate on a tunnel is collected at regular intervals that are configured by using the **application** command in MPLS-TE auto bandwidth interface configuration mode. When the application period timer expires, and when the difference between the measured and the current bandwidth exceeds the adjustment threshold, the tunnel is re-optimized. Then, the bandwidth samples are cleared to record the new largest output rate at the next interval. If a tunnel is shut down, and is later brought again, the adjusted bandwidth is lost, and the tunnel is brought back with the initially configured bandwidth. When the tunnel is brought back, the application period is reset.

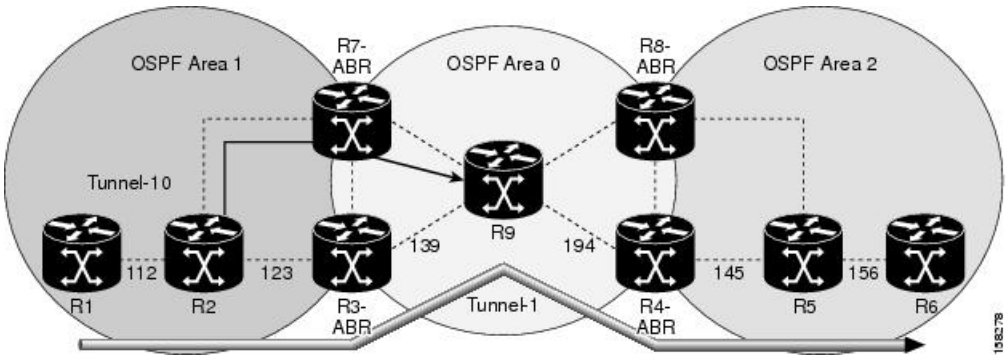
MPLS Traffic Engineering Interarea Tunneling

The MPLS-TE interarea tunneling feature allows you to establish TE tunnels spanning multiple Interior Gateway Protocol (IGP) areas and levels, thus eliminating the requirement that headend and tailend routers reside in a single area.

Interarea support allows the configuration of a TE LSP that spans multiple areas, where its headend and tailend label switched routers (LSRs) reside in different IGP areas. Customers running multiple IGP area backbones (primarily for scalability reasons) requires Multiarea and Interarea TE . This lets you limit the amount of flooded information, reduces the SPF duration, and lessens the impact of a link or node failure within an area, particularly with large WAN backbones split in multiple areas.

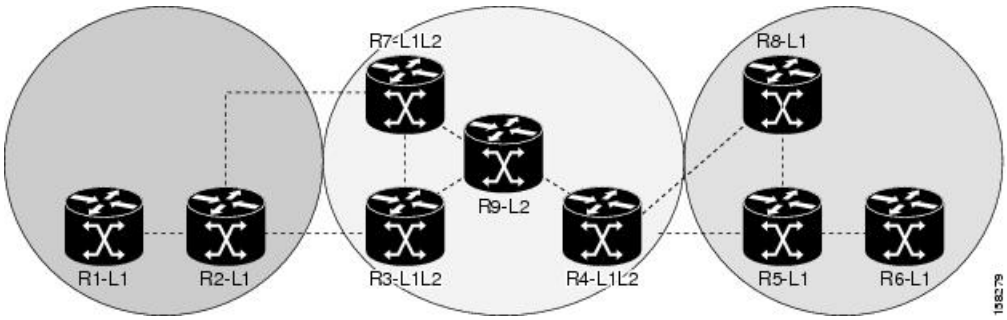
The following figure shows a typical interarea TE network using OSPF.

Figure 18: Interarea (OSPF) TE Network Diagram



The following figure shows a typical interlevel (IS-IS) TE Network.

Figure 19: Interlevel (IS-IS) TE Network Diagram



As shown in the Figure 19: Interlevel (IS-IS) TE Network Diagram, on page 135, R2, R3, R7, and R4 maintain two databases for routing and TE information. For example, R3 has TE topology information related to R2, flooded through Level-1 IS-IS LSPs plus the TE topology information related to R4, R9, and R7, flooded as Level 2 IS-IS Link State PDUs (LSPs) (plus, its own IS-IS LSP).

Loose hop optimization allows the re-optimization of tunnels spanning multiple areas and solves the problem which occurs when an MPLS-TE LSP traverses hops that are not in the LSP's headend's OSPF area and IS-IS level. Interarea MPLS-TE allows you to configure an interarea traffic engineering (TE) label switched path (LSP) by specifying a loose source route of ABRs along the path. Then it is the responsibility of the ABR (having a complete view of both areas) to find a path obeying the TE LSP constraints within the next area to reach the next hop ABR (as specified on the headend router). The same operation is performed by the last ABR connected to the tailend area to reach the tailend LSR.

You must be aware of these considerations when using loose hop optimization:

- You must specify the router ID of the ABR node (as opposed to a link address on the ABR).
- When multiarea is deployed in a network that contains subareas, you must enable MPLS-TE in the subarea for TE to find a path when loose hop is specified.
- You must specify the reachable explicit path for the interarea tunnel.

UCMP Over MPLS-TE

In Equal Cost Multi Path (ECMP), you can load-balance routed traffic over multiple paths of the same cost. With Unequal-Cost Multipath (UCMP), you can load-balance traffic over multiple paths of varying costs.

Consider three forwarding links, with two 10 Gigabit Ethernet links and a 100 Gigabit Ethernet link, as shown in the image.

In such a scenario, the incoming traffic load is not equally distributed using ECMP. On the other hand, UCMP applies a weight to a path, and adds more forwarding instances to a path that has a higher weight (larger bandwidth). This results in an equal load distribution over paths of varying bandwidths (and costs).

Configuration Example

UCMP Configuration:

```
R1# configure
R1(config)# mpls traffic-eng
R1(config-mpls-te)# load-share unequal
R1(config-mpls-te)# commit
```

Tunnels Configuration:

```
R1(config)# interface tunnel-te 1
R1(config-if)# ipv4 unnumbered loopback 1
R1(config-if)# load-share 5
R1(config-if)# autoroute announce
R1(config-if-tunte-aa)# commit
R1(config-if-tunte-aa)# exit
R1(config-if)# destination 172.16.0.1
R1(config-if)# path-option 1 dynamic

R1(config)# interface tunnel-te 2
R1(config-if)# ipv4 address 192.168.0.1 255.255.255.0
R1(config-if)# load-share 6
R1(config-if)# autoroute announce
R1(config-if-tunte-aa)# commit
R1(config-if-tunte-aa)# exit
R1(config-if)# destination 172.16.0.1
R1(config-if)# path-option 1 dynamic
```

Associated Commands

- load-share
- load-share unequal
- show cef

Verification

Verify UCMP Configuration:

```
R1# show cef 172.16.0.1 detail

172.16.0.1/32, version 16, internal 0x1000001 0x0 (ptr 0x97de1a58) [1], 0x0 (0x97fa3728),
0xa20 (0x98fc00a8)
 Updated Jun 17 16:07:46.325
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
  gateway array (0x97e0ba08) reference count 3, flags 0x68, source lsd (5), 1 backups
                [3 type 4 flags 0x8401 (0x9849f728) ext 0x0 (0x0)]
  LW-LDI[type=1, refc=1, ptr=0x97fa3728, sh-ldi=0x9849f728]
  gateway array update type-time 1 Jun 17 16:07:46.325
 LDI Update time Jun 17 16:07:46.350
```

```
 LW-LDI-TS Jun 17 16:07:46.350
   via 172.16.0.1/32, tunnel-te1, 5 dependencies, weight 100, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0x98e19380 0x98e192f0]
    next hop 172.16.0.1/32
    local adjacency
     local label 24001      labels imposed {ImplNull}
   via 172.16.0.1/32, tunnel-te2, 7 dependencies, weight 10, class 0 [flags 0x0]
    path-idx 1 NHID 0x0 [0x98e194a0 0x98e19410]
    next hop 172.16.0.1/32
    local adjacency
     local label 24001      labels imposed {ImplNull}

    Weight distribution:

    slot 0, weight 100, normalized_weight 10, class 0
    slot 1, weight 10, normalized_weight 1, class 0
    Load distribution: 0 0 0 0 0 0 0 0 0 0 1 (refcount 3)

    Hash  OK  Interface                 Address
    0     Y   tunnel-te1                point2point
    1     Y   tunnel-te1                point2point
    2     Y   tunnel-te1                point2point
    3     Y   tunnel-te1                point2point
    4     Y   tunnel-te1                point2point
    5     Y   tunnel-te1                point2point
    6     Y   tunnel-te1                point2point
    7     Y   tunnel-te1                point2point
    8     Y   tunnel-te1                point2point
    9     Y   tunnel-te1                point2point
    10    Y   tunnel-te2                point2point
```

Some sample output:

```
Router# show run formal mpls traffic-eng

mpls traffic-eng
mpls traffic-eng interface Bundle-Ether2
mpls traffic-eng interface Bundle-Ether3
..
mpls traffic-eng load-share unequal
mpls traffic-eng reoptimize 180
mpls traffic-eng signalling advertise explicit-null
mpls traffic-eng reoptimize timers delay path-protection 60

Router# show run formal interface tunnel-te400

interface tunnel-te400
interface tunnel-te400 description TE-STI-GRTMIABR5-GRTBUEBA3-BW-0
interface tunnel-te400 ipv4 unnumbered Loopback0
interface tunnel-te400 load-interval 30
interface tunnel-te400 signalled-name TE-STI-GRTMIABR5-GRTBUEBA3-BW-0
interface tunnel-te400 load-share 80
interface tunnel-te400 autoroute destination 94.142.100.214
interface tunnel-te400 destination 94.142.100.214
interface tunnel-te400 path-protection
interface tunnel-te400 path-option 1 explicit name BR5-BA3-0 protected-by 2
interface tunnel-te400 path-option 2 explicit name BW-BR5-1-VAP3-BA3-0
interface tunnel-te400 path-option 3 explicit name BW-BR5-VAP3-BA3-0

Router# show run formal interface tunnel-te406

interface tunnel-te406
interface tunnel-te406 description TE-STI-GRTMIABR5-GRTBUEBA3-BW-20
interface tunnel-te406 ipv4 unnumbered Loopback0
```

```
interface tunnel-te406 load-interval 30
interface tunnel-te406 signalled-name TE-STI-GRTMIABR5-GRTBUEBA3-BW-20
interface tunnel-te406 load-share 10
interface tunnel-te406 autoroute destination 94.142.100.214
interface tunnel-te406 destination 94.142.100.214
interface tunnel-te406 path-protection
interface tunnel-te406 path-option 1 explicit name BR5-1-BA3-0 protected-by 2
interface tunnel-te406 path-option 2 explicit name BW-BR5-VAP4-BA3-0
```

```
Router# show cef 94.142.100.214/32 det

94.142.100.214/32, version 25708656, attached, internal 0x4004081 0x0 (ptr 0x764ff1b0) [3],
 0x0 (0x7267d848), 0x440 (0x7d93b2b8)
 Updated Nov 19 08:02:26.545
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
  gateway array (0x72411528) reference count 3, flags 0xd0, source lsd (4), 1 backups
                [3 type 4 flags 0x10101 (0x7300d648) ext 0x0 (0x0)]
  LW-LDI[type=1, refc=1, ptr=0x7267d848, sh-ldi=0x7300d648]
   via tunnel-te400, 3 dependencies, weight 80, class 0 [flags 0x8]
    path-idx 0 NHID 0x0 [0x72082a40 0x72983b58]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te406, 3 dependencies, weight 10, class 0 [flags 0x8]
    path-idx 1 NHID 0x0 [0x7207b4ac 0x729886bc]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te410, 3 dependencies, weight 80, class 0 [flags 0x8]
    path-idx 2 NHID 0x0 [0x72085218 0x72985ee4]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te426, 3 dependencies, weight 10, class 0 [flags 0x8]
    path-idx 3 NHID 0x0 [0x7207d4b4 0x7297fecc]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te427, 3 dependencies, weight 25, class 0 [flags 0x8]
    path-idx 4 NHID 0x0 [0x720802cc 0x7298726c]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te1089, 3 dependencies, weight 40, class 0 [flags 0x8]
    path-idx 5 NHID 0x0 [0x72081848 0x7298037c]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te1090, 3 dependencies, weight 60, class 0 [flags 0x8]
    path-idx 6 NHID 0x0 [0x7207d770 0x72987780]
    local adjacency
     local label 16440      labels imposed {ImplNull}
   via tunnel-te1099, 3 dependencies, weight 60, class 0 [flags 0x8]
    path-idx 7 NHID 0x0 [0x7207ed50 0x72981c7c]
    local adjacency
     local label 16440      labels imposed {ImplNull}

    Weight distribution:
    slot 0, weight 80, normalized_weight 7, class 0
    slot 1, weight 10, normalized_weight 1, class 0
    slot 2, weight 80, normalized_weight 7, class 0
    slot 3, weight 10, normalized_weight 1, class 0
    slot 4, weight 25, normalized_weight 1, class 0
    slot 5, weight 40, normalized_weight 3, class 0
    slot 6, weight 60, normalized_weight 5, class 0
    slot 7, weight 60, normalized_weight 5, class 0
```

```
Router# show cef 94.142.100.213/32 det

94.142.100.213/32, version 25708617, attached, internal 0x4004081 0x0 (ptr 0x771925c8) [3],
 0x0 (0x7267a594), 0x440 (0x7d93d364)
```

```
Updated Nov 19 08:02:01.029
Prefix Len 32, traffic index 0, precedence n/a, priority 3
 gateway array (0x7240f638) reference count 3, flags 0xd0, source lsd (4), 1 backups
              [3 type 4 flags 0x10101 (0x73013360) ext 0x0 (0x0)]
 LW-LDI[type=1, refc=1, ptr=0x7267a594, sh-ldi=0x73013360]
  via tunnel-te220, 3 dependencies, weight 60, class 0 [flags 0x8]
   path-idx 0 NHID 0x0 [0x7207d838 0x72982af0]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te230, 3 dependencies, weight 60, class 0 [flags 0x8]
   path-idx 1 NHID 0x0 [0x7207d068 0x72986e20]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te236, 3 dependencies, weight 50, class 0 [flags 0x8]
   path-idx 2 NHID 0x0 [0x720830e4 0x7297f508]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te246, 3 dependencies, weight 100, class 0 [flags 0x8]
   path-idx 3 NHID 0x0 [0x7207a1ec 0x7298483c]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te221, 3 dependencies, weight 50, class 0 [flags 0x8]
   path-idx 4 NHID 0x0 [0x7207ea30 0x72982834]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te222, 3 dependencies, weight 25, class 0 [flags 0x8]
   path-idx 5 NHID 0x0 [0x72084a48 0x72989850]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te1091, 3 dependencies, weight 30, class 0 [flags 0x8]
   path-idx 6 NHID 0x0 [0x720851b4 0x729895f8]
   local adjacency
    local label 17561      labels imposed {ImplNull}
  via tunnel-te342, 3 dependencies, weight 100, class 0 [flags 0x8]
   path-idx 7 NHID 0x0 [0x72085344 0x7298b024]
   local adjacency
    local label 17561      labels imposed {ImplNull}

   Weight distribution:
   slot 0, weight 60, normalized_weight 2, class 0
   slot 1, weight 60, normalized_weight 2, class 0
   slot 2, weight 50, normalized_weight 2, class 0
   slot 3, weight 100, normalized_weight 4, class 0
   slot 4, weight 50, normalized_weight 2, class 0
   slot 5, weight 25, normalized_weight 1, class 0
   slot 6, weight 30, normalized_weight 1, class 0
   slot 7, weight 100, normalized_weight 4, class 0

   Load distribution: 0 0 1 1 2 2 3 3 3 3 4 4 5 6 7 7 7 7 (refcount 3)

   Hash  OK  Interface                 Address
   0     Y   tunnel-te220              point2point
   1     Y   tunnel-te220              point2point
   2     Y   tunnel-te230              point2point
   3     Y   tunnel-te230              point2point
   4     Y   tunnel-te236              point2point
   5     Y   tunnel-te236              point2point
   6     Y   tunnel-te246              point2point
   7     Y   tunnel-te246              point2point
   8     Y   tunnel-te246              point2point
   9     Y   tunnel-te246              point2point
   10    Y   tunnel-te221              point2point
   11    Y   tunnel-te221              point2point
   12    Y   tunnel-te222              point2point
```

```
13    Y   tunnel-te1091            point2point
14    Y   tunnel-te342             point2point
15    Y   tunnel-te342             point2point
16    Y   tunnel-te342             point2point
17    Y   tunnel-te342             point2point
```

Event history for MPLS-TE tunnels

MPLS-TE tunnel event histories are network diagnostic records that

- chronologically document significant events and actions that occur within MPLS-TE headend tunnels
- enable detailed tracking and analysis of tunnel lifecycle changes, including RSVP signaling events, and
- facilitate monitoring, troubleshooting, and compliance by recording each event with timestamps and associated metadata.

Table 24: Feature History Table

Feature Name	Release	Feature Description
Event history for MPLS-TE headend tunnels	Release 25.4.1	Introduced in this release on: NCS 5500 fixed port routers; NCS 5700 fixed port routers; NCS 5500 modular routers (NCS 5500 line cards; NCS 5700 line cards [Mode: Compatibility; Native]) This feature introduces a granular and configurable event tracking mechanism for operational visibility into MPLS-TE tunnels. You can enable event history with the **mpls traffic-eng event-history tunnel** command and set the number of tracked events with the **mpls traffic-eng event-history tunnel event-count** command. This feature introduces these changes: CLI: • Three new keywords, **event-history**, **tunnel**,and **event-count**, are added to the **mpls traffic-eng** command. • Two new keywords, **event-history**, and **tunnels** are added to the **show mpls traffic-eng** command. • A new **clear mpls traffic-eng event-history tunnels** command has been added. • The **show mpls traffic-eng counters global** and **show mpls traffic-eng chunks** commands have been modified to include the event history information.

Cisco IOS XR Release 25.4.1 introduces the tunnel event history for MPLS-TE tunnels to track and analyse the timeline of events in the tunnel lifecycle, including RSVP signaling. This feature enables enhanced operational visibility for MPLS-TE tunnels by providing granular, configurable event tracking.

To enable the MPLS-TE tunnel event history feature, use the **mpls traffic-eng event-history tunnel** command. To set the event count for the tunnel event history, configure the use the **mpls traffic-eng event-history tunnel event-count** command.

Benefits of tunnel event history

You can use the event history feature for MPLS-TE headend tunnels to:

- maintain a familiar workflow with minimal changes,
- log events in dedicated, per-tunnel buffers for fair and isolated event recording,
- flexibly capture and report events with comprehensive argument handling,
- compress duplicate events to minimize log size and clutter, and
- enhance debugging with accessible historical records for troubleshooting.

Restrictions for tunnel event history

The tunnel event history feature is:

- Disabled by default.
- Supported only on MPLS-TE headend tunnels.

Configure tunnel event history and event count

This procedure helps you to configure the tunnel event history and the event count.

Procedure

Step 1 Enable the tunnel event history.

Example:

```
Router# configure
Router(config)# mpls traffic-eng event-history tunnel
Router(config-te-evt-hist)# commit
```

Step 2 Specify the event count.

Example:

```
Router# configure
Router(config)# mpls traffic-eng event-history tunnel event-count 45
Router(config-te-evt-hist)# commit
```

The default number of events is 25. You can set up to 1000 events per tunnel.

Step 3 Verify the tunnel event history configuration.

Example:

This output shows the event history for all tunnels.

```
Router# show mpls traffic-eng event-history tunnel all
Wed Dec 3 14:08:59.529 UTC
```

```
Name: tunnel-te1
Event history (oldest first):
ID                   Time                LSP  Event-Type  Rep  Event-Message
01764770758196406124 Dec 3 14:06:05.309   2  Path          0  BW:300 kbps ERO: 10.1.3.103 10.3.5.105
 10.5.6.106 20.1.0.106
01764770758196406128 Dec 3 14:06:05.346   2  Path Err      0  (24,0), Error: routing (24), Suberror:
 none (0), error-node: 10.1.3.103
01764770758196406129 Dec 3 14:06:05.346   2  Path Tear     0
01764770758196406131 Dec 3 14:06:08.349   3  Path          0  BW:300 kbps ERO: 10.1.3.103 10.3.5.105
 10.5.6.106 20.1.0.106
01764770758196406132 Dec 3 14:06:08.449   3  Resv          0   BW:300 kbps RRO:
01764770758196406134 Dec 3 14:06:08.449   3  Oper State    0   New tunnel state: UP

Name: T10
Event history (oldest first):
ID                   Time                LSP  Event-Type  Rep  Event-Message
01764770758196406125 Dec 3 14:06:05.309   2  Path          0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104
 10.4.6.106 20.1.0.106
01764770758196406126 Dec 3 14:06:05.346   2  Path Err      0  (24,0), Error: routing (24), Suberror:
 none (0), error-node: 10.1.2.102
01764770758196406127 Dec 3 14:06:05.346   2  Path Tear     0
01764770758196406130 Dec 3 14:06:08.346   3  Path          0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104
 10.4.6.106 20.1.0.106
01764770758196406133 Dec 3 14:06:08.449   3  Resv          0  BW:300 kbps RRO:
01764770758196406135 Dec 3 14:06:08.496   3  Oper State    0  New tunnel state: UP

Name: SELFPING-101
 Event history (oldest first):
 ID                   Time                  LSP  Event-Type  Rep  Event-Message
 01769076278173248001  Jan 22 05:30:13.487    2  Path         0  BW:0 kbps ERO: 10.10.10.2 17.17.17.2
 17.17.17.3 22.22.22.3 22.22.22.4 192.168.0.4
 01769076278173248002  Jan 22 05:30:13.546    2  Resv         0  BW:0 kbps RRO:
 01769076278173248003  Jan 22 05:30:23.551    2  Self-Ping    0  Timed-out (started at Jan 22
05:01:30.769)
 01769076278173248004  Jan 22 05:30:23.554    2  Oper State   0  New tunnel state: UP

Name: SELFPING-103
 Event history (oldest first):
 ID                   Time                  LSP  Event-Type  Rep  Event-Message
 01769076278173248011  Jan 22 05:31:54.518    2  Path         0  BW:0 kbps ERO: 10.10.10.2 17.17.17.2
 17.17.17.3 22.22.22.3 22.22.22.4 192.168.0.4
 01769076278173248012  Jan 22 05:31:54.581    2  Resv         0  BW:0 kbps RRO:
 01769076278173248013  Jan 22 05:32:02.917    2  Self-Ping    0  Aborted (started at Jan 22
05:01:30.769)
 01769076278173248014  Jan 22 05:32:02.925    2  Path Tear    0
 01769076278173248015  Jan 22 05:32:02.936    3  Path         0  BW:0 kbps ERO: 11.11.11.2 17.17.17.2
 17.17.17.3 22.22.22.3 22.22.22.4 192.168.0.4
 01769076278173248016  Jan 22 05:32:02.987    3  Resv         0  BW:0 kbps RRO:
 01769076278173248017  Jan 22 05:32:06.994    3  Self-Ping    0  Succeeded (started at Jan 22
05:01:30.769)
 01769076278173248018  Jan 22 05:32:06.995    3  Oper State   0  New tunnel state: UP
```

Example:

This output shows the event history for a specific tunnel.

```
Router# show mpls traffic-eng event-history tunnels tunnel-T10


Thu Dec 18 07:30:57.167 UTC
Name: tunnel-T10
 Event history (oldest first):
 ID                   Time                  LSP  Event-Type  Rep  Event-Message
 01765952238563713933  Jan 1 07:15:28.220    5  Path Tear    0
```

```
 01765952238563713934  Jan 1 07:15:32.660    8  Path        0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104

 01765952238563713935  Jan 1 07:15:32.763     8  Resv          0  BW:300 kbps RRO:
 01765952238563713936  Jan 1 07:15:32.803     6  Path Tear     0
 01765952238563713937  Jan 1 07:15:37.387    9  Path        0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104

 01765952238563713938  Jan 1 07:15:37.480     9  Resv          0  BW:300 kbps RRO:
 01765952238563713939  Jan 1 07:15:37.517     7  Path Tear     0
 01765952238563713940  Jan 1 07:15:47.193   10  Path         0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104

 01765952238563713941  Jan 1 07:15:47.283    10  Resv          0  BW:300 kbps RRO:
 01765952238563713942  Jan 1 07:15:47.320     8  Path Tear     0
 01765952238563713943  Jan 1 07:16:07.420     9  Path Tear     0
 01765952238563713944  Jan 1 07:16:15.573   11  Path         0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104

 01765952238563713945  Jan 1 07:16:15.670    11  Resv          0  BW:300 kbps RRO:
 01765952238563713946  Jan 1 12:23:05.711    10  Self-Ping     0  Succeeded (started at Jan  1
12:23:05.709)
 01765952238563713947  Jan 1 07:16:35.810    10  Path Tear     0
 01765952238563713948  Jan 1 07:22:49.863   12  Path         0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104

 01765952238563713949  Jan 1 07:22:49.963    12  Resv          0  BW:300 kbps RRO:
 01765952238563713950  Jan 1 07:23:10.103    11  Path Tear     0
 01765952238563713951  Jan 1 07:23:49.240   13  Path         0  BW:300 kbps ERO: 10.1.2.102 10.2.4.104

 01765952238563713952  Jan 1 07:23:49.340    13  Resv          0  BW:300 kbps RRO:
 01765952238563713953  Jan 1 07:24:09.480    12  Path Tear     0
 01765952238563713954  Jan 1 12:45:45.473    11  Self-Ping     0  Succeeded (started at Jan  1
12:45:45.472)
```

Example:

This output shows the specified event count.

```
Router# show run mpls traffic-eng event-history
Thu Dec 4 06:56:44.650 UTC
mpls traffic-eng
event-history
tunnel
event-count 45
!
!
!
```

Example:

This example shows the event history size.

```
Router# show mpls traffic-eng counters global

Wed Dec 3 14:09:38.405 UTC

Global Statistics:
Number of VIF : 2
Number of LSP : 2
Number of S2L : 2

Checkpoint request queue high watermark : 3
Checkpoint response queue high watermark : 16
Checkpoint reschedules : 28
Checkpoint enqueues (request queue) : 86
Checkpoint dequeues (request queue) : 86
Checkpoint enqueues (response queue) : 86
Checkpoint dequeues (response queue) : 86
Checkpoint write (VIF) : 32
Checkpoint read (VIF) : 0
```

```
Checkpoint write (DST) : 0
Checkpoint read (DST) : 0
Checkpoint write (S2L) : 4
Checkpoint read (S2L) : 0
Checkpoint write (Autobackup) : 0
Checkpoint read (Autobackup) : 0
Checkpoint write (PI backup) : 0
Checkpoint read (PI backup) : 0
Number of RSVP connection : 1
Number of RSVP disconnection : 0
Number of RSVP restart time change : 0
Number of LSD connection : 1
Number of LSD disconnection : 0
Number of LSD replay : 1
Number of LMRIB connection : 1
Number of LMRIB disconnection : 0
Number of LMRIB restart time change : 1
Number of LMRIB restart failed : 0
Number of LMRIB restart passed : 1
Number of LMRIB recovery failed : 0
Number of LMRIB recovery passed : 1
Number of RSI connection : 1
Number of RSI disconnection : 0
Event-history size (bytes)
 Tunnels (excluding chunks) : 16444912
```

Example:

This example displays the memory information for the tunnel event history.

```
Router# show mpls traffic-eng chunks

  Tue Dec  9 06:30:40.055 UTC
Pool name            InUse     Free      Allocs    Deallocs  Sib creat Sib trimm
-------------------- --------- --------- --------- --------- --------- ---------
mpls wavl Chunks             0     32000         0         0         0         0
mpls wavl Chunks            28      7972        28         0         0         0
mpls wavl Chunks           101      3899       101         0         0         0
mpls wavl Chunks            59      1941        59         0         0         0
mpls wavl Chunks            27       973        27         0         0         0
mpls wavl Chunks             9       491         9         0         0         0
mpls wavl Chunks             0       240         0         0         0         0
mpls wavl Chunks             0       120         0         0         0         0
TE S2L chunk                 0       200         0         0         0         0
TE S2L stats chunk           0      1024         0         0         0         0
TE LSP chunk                 0       200         0         0         0         0
TE collab chunk             15        25        15         0         0         0
TE FSM Q chunk               0       128         0         0         0         0
TE NextHop Route Po          0       128         0         0         0         0
TE Binding Sid Cont          0       128         0         0         0         0
TE CHKPT Q Elements          0       600        10        10         0         0
RSVP CHKPT LIST Chu          0      8192         0         0         0         0
TE Tunnel Service R          0       128         0         0         0         0
TE Tunnel Service R          0       128         0         0         0         0
RSVP API CTX                 0       156         0         0         0         0
RSVP API CTX stats           0        48         0         0         0         0
RSVP API CTX event           0      1248         0         0         0         0
RSVP API CTX event           0      1248         0         0         0         0
RSVP API CTX Histor          0      1248         0         0         0         0
TE S2L RW chunk              0       124         0         0         0         0
TE S2L RW chunk              0       128         0         0         0         0
TE Event History             0       200         0         0         0         0
VIF Pulse Context c          0      1024       668       668         0         0
```

Step 4 (Optional) Clear the tunnel event history for all tunnels, if required.

Example:

```
Router# clear mpls traffic-eng event-history tunnels all
```

Configuring Performance Measurement

Network performance metrics such as packet loss, delay, delay variation, and bandwidth utilization is a critical measure for traffic engineering (TE) in service provider networks. These network performance metrics provide network operators information about the performance characteristics of their networks for performance evaluation and helps to ensure compliance with service level agreements. The service-level agreements (SLAs) of service providers depend on the ability to measure and monitor these network performance metrics. Network operators can use performance measurement (PM) feature to monitor the network metrics for links as well as end-to-end TE label switched paths (LSPs).

Path Calculation Metric Type

To configure the metric type to be used for path calculation for a given tunnel, use the **path-selection metric** command in either the MPLS-TE configuration mode or under the tunnel interface configuration mode.

The metric type specified per interface takes the highest priority, followed by the MPLS-TE global metric type.

Note If the delay metric is configured, CSPF finds a path with optimized*minimum* link delay metric. See the *Configuring Performance Measurement* chapter in the Segment Routing Configuration Guide for information on configuring interface performance delay measurement.

Configuration Example

The following example shows how to set the path-selection metric to use the IGP metric under a specific tunnel interface:

```
Router# configure
Router(config)# interface tunnel-te 1
Router(config-if)# path-selection metric igp
Router(config-if)# commit
```

The following example shows how to set the path-selection metric to use the delay metric under the MPLS-TE configuration mode:

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# path-selection metric delay
Router(config-mpls-te)# commit
```

Path-Selection Delay Limit

Apply the **path-selection delay-limit** configuration to set the upper limit on the path aggregate delay when computing paths for MPLS-TE LSPs. After you configure the **path-selection delay-limit** value, if the sum of minimum-delay metric from all links that are traversed by the path exceeds the specified delay-limit, CSPF will not return any path. The periodic path verification checks if the delay-limit is crossed.

The **path-selection delay-limit** value can be configured at the global MPLS-TE, per-interface tunnel, and per path-option attribute set. The path-selection delay-limit per path-option attribute set takes the highest priority, followed by per-interface, and then the MPLS-TE global path-selection delay-limit values.

The delay limit range is a value from 1 to 4294967295 microseconds.



Note See the *Configuring Performance Measurement* chapter in the Segment Routing Configuration Guide for information on configuring interface performance delay measurement.

Configuration Example

The following example shows how to set the path-selection delay limit under a specific tunnel interface:

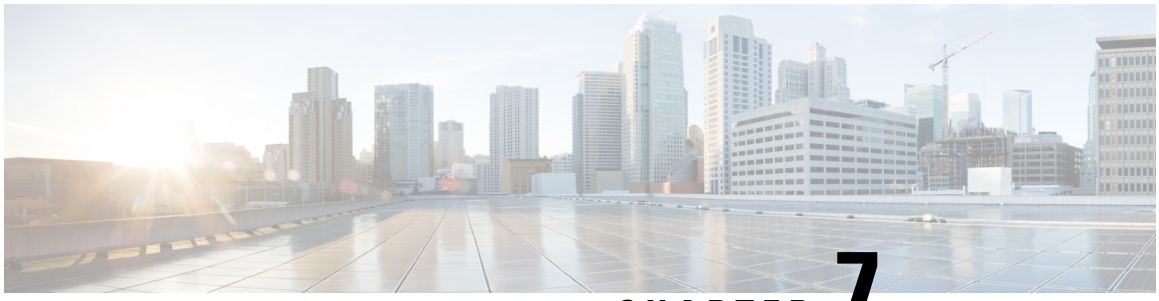
```
Router# configure
Router(config)# interface tunnel-te2000
Router(config-if)# path-selection metric delay
Router(config-if)# path-selection delay-limit 200
Router(config-if)# commit
```

The following example shows how to set the path-selection delay limit under a path-option attribute set:

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# attribute-set path-option test
Router(config-te-attribute-set)# path-selection delay-limit 300
Router(config-te-attribute-set)# root
Router(config)# interface tunnel-te1000
Router(config-if)# path-option 10 dynamic attribute-set test
Router(config-if)# commit
```

The following example shows how to set the path-selection delay limit under the global MPLS-TE configuration mode:

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# path-selection metric delay
Router(config-mpls-te)# path-selection delay-limit 150
Router(config-mpls-te)# commit
```



CHAPTER 7

Implementing MPLS OAM

- Implementing MPLS OAM, on page 149
- MPLS OAM Using Nil FEC, on page 152

Implementing MPLS OAM

MPLS Operations, Administration, and Maintenance (OAM) helps service providers to monitor label-switched paths (LSPs) and quickly isolate MPLS forwarding problems to assist with fault detection and troubleshooting in an MPLS network. This module describes MPLS LSP Ping and Traceroute features which can be used for failure detection and troubleshooting of MPLS networks.

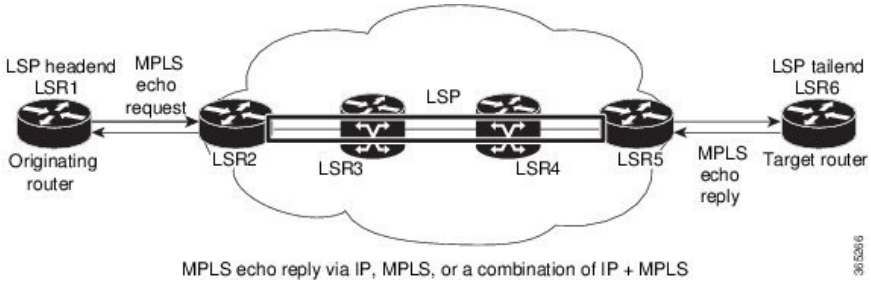
MPLS LSP Ping

The MPLS LSP Ping feature is used to check the connectivity between Ingress LSR and egress LSRs along an LSP. MPLS LSP ping uses MPLS echo request and reply messages, similar to Internet Control Message Protocol (ICMP) echo request and reply messages, to validate an LSP. While ICMP echo request and reply messages validate IP networks, MPLS echo and reply messages validate MPLS networks. 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 itself. 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 and it prevents the IP packet from being IP switched 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 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.

The following figure shows MPLS LSP ping echo request and echo reply paths.

Figure 20: MPLS LSP Ping Echo Request and Reply Paths



Configuration Examples

This example shows how to use MPLS LSP ping to test the connectivity of an IPv4 LDP LSP. The destination is specified as a Label Distribution Protocol (LDP) IPv4 prefix and Forwarding Equivalence Class (FEC) type is specified as generic.

```
RP/0/RP0/CPU0:router# ping mpls ipv4 10.1.1.2/32 fec-type generic

Wed Nov 25 03:36:33.143 UTC
Sending 5, 100-byte MPLS Echos to 10.1.1.2/32,
      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, '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 = 2/2/3 ms
```

This example shows how to use MPLS LSP ping to test the connectivity when the destination is specified as a MPLS traffic engineering (TE) tunnel.

```
RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-te 4003 source 10.1.1.2

Tue Nov 24 20:39:39.179 PST

Sending 5, 100-byte MPLS Echos to tunnel-te4003,
      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, '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 = 3/3/4 ms
```

This example shows how to use the **show mpls oam** command to display the MPLS OAM information

```
RP/0/RP0/CPU0:router# show mpls oam counters packet

Wed Nov 25 03:38:07.397 UTC Global Packet Statistics:

                       Pkt                 Bytes
                       ---------------     ---------------
Receive Counts:
     Good Requests:                    0                   0
      Good Replies:                   10                 760
 Unknown Pkt Types:                    0                   0
   IP header error:                    0                   0
  UDP header error:                    0                   0
             Runts:                    0                   0
  Dropped (Q full):                    0                   0
     General error:                    0                   0
      Error, no IF:                    0                   0
  Error, no memory:                    0                   0

Transmit Counts:
              Good:                   10                 960
           Dropped:                    0                   0
```

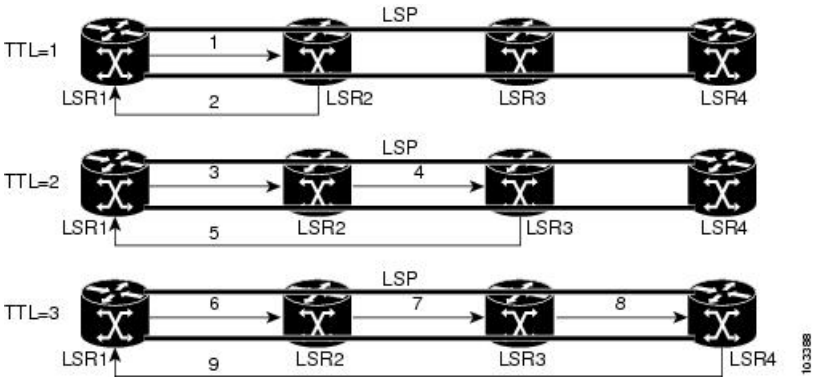
MPLS LSP Traceroute

The MPLS LSP Traceroute feature is used to isolate the failure point of an LSP. It is used for hop-by-hop fault localization and path tracing. The MPLS LSP Traceroute feature relies on the expiration of the Time to Live (TTL) value of the packet that carries the echo request. When the MPLS echo request message hits a transit node, it checks the TTL value and if it is expired, the packet is passed to the control plane, else the message is forwarded. If the echo message is passed to the control plane, a reply message is generated based on the contents of the request message.

Note When any node receives an MPLS packet with a TTL value as 0, it is dropped as it is an invalid packet.

The following figure shows an MPLS LSP traceroute example with an LSP from LSR1 to LSR4.

Figure 21: MPLS LSP Traceroute



Guidelines and restrictions

MPLS traceroute may fail with a `DS Map mismatch` error in networks with multi-level Equal-Cost Multi-Path (ECMP) topologies. This occurs because the load-balancing decisions made by the IOS XR software and the Network Processing Unit (NPU) hardware may not align. As a result, consistency between the predicted and actual paths in the ECMP environment cannot be guaranteed.

Configuration Examples

This example shows how to use the **traceroute** command to trace to a destination with Forwarding Equivalence Class (FEC) type specified as generic.

```
RP/0/RP0/CPU0:router# traceroute mpls ipv4 192.168.0.1/32 fec-type generic
Mon Nov 30 17:48:45.585 UTC

Tracing MPLS Label Switched Path to 192.168.0.1/32, 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, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

  0 10.1.1.57 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.1.1.58 7 ms23:19
```

MPLS OAM Using Nil FEC

The Nil-FEC LSP ping and traceroute operations are extensions of regular MPLS ping and traceroute. MPLS ping and traceroute requires at least one forwarding equivalence class (FEC) in the target FEC stack. In Nil-FEC ping and traceroute operations, an explicit FEC is not associated with the label. Nil-FEC LSP ping and traceroute support MPLS static LSPs and also act as an additional diagnostic tool for all other LSP types. Nil-FEC LSP ping and traceroute allow network operators to provide the ability to freely test any label stack by allowing them to specify the following:

- label stack
- outgoing interface
- nexthop address

The following table shows the syntax for the ping and traceroute commands.

Table 25: LSP Ping and Traceroute Nil FEC Commands

Command Syntax
ping mpls nil-fec labels {*label*[**,***label*]} [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]]
traceroute mpls nil-fec labels {*label*[**,***label*]} [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]]

Examples: LSP Ping Nil FEC and LSP Traceroute Nil FEC

The examples in this section use the following topology:

```
Node loopback IP address: 172.18.1.3   172.18.1.4   172.18.1.5   172.18.1.7
Node label:               16003        16004        16005        16007
Nodes:                    Arizona ---- Utah ------- Wyoming ---- Texas

Interface:            GigabitEthernet0/2/0/1   GigabitEthernet0/2/0/1
Interface IP address:         10.1.1.3              10.1.1.4
```

```
RP/0/RP0/CPU0:router-arizona# show mpls forwarding

Tue May  2 13:44:31.999 EDT
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
16004  Pop         No ID              Gi0/2/0/1    10.1.1.4        1392
       Pop         No ID              Gi0/2/0/2    10.1.2.2        0
16005  16005       No ID              Gi0/2/0/0    10.1.1.4        0
       16005       No ID              Gi0/2/0/1    10.1.2.2        0
16007  16007       No ID              Gi0/2/0/0    10.1.1.4        4752
       16007       No ID              Gi0/2/0/1    10.1.2.2        0
```

This example shows how to use Nil-FEC LSP ping to test a label stack.

```
RP/0/RP0/CPU0:router-arizona# ping mpls nil-fec labels 16005,16007 output interface
GigabitEthernet 0/2/0/1 nexthop 10.1.1.4 repeat 1
Sending 1, 72-byte MPLS Echos with Nil FEC labels 16005,16007,
     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,
  'd' - see DDMAP for return code,
  '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 = 1/1/1 ms
 Total Time Elapsed 0 ms
```

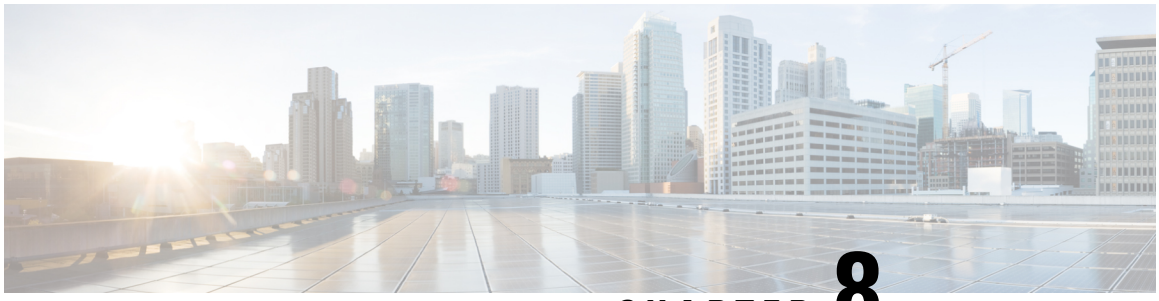
This example shows how to use Nil-FEC LSP traceroute for a label stack.

```
RP/0/RP0/CPU0:router-arizona# traceroute mpls nil-fec labels 16005,16007 output interface
GigabitEthernet 0/2/0/1 nexthop 10.1.1.4
Tracing MPLS Label Switched Path with Nil FEC labels 16005,16007, 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,
  'd' - see DDMAP for return code,
```

```
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.
  0 10.1.1.3 MRU 1500 [Labels: 16005/16007/explicit-null Exp: 0/0/0]
L 1 10.1.1.4 MRU 1500 [Labels: implicit-null/16007/explicit-null Exp: 0/0/0] 1 ms
L 2 10.1.1.5 MRU 1500 [Labels: implicit-null/explicit-null Exp: 0/0] 1 ms
! 3 10.1.1.7 1 ms
```



CHAPTER 8

Implementing GMPLS UNI

Table 26: Feature History Table

Feature Name	Release Information	Feature Description
GMPLS UNI	Release 7.3.1	The Generalized Multiprotocol Label Switching (GMPLS) User Network Interface (UNI) creates a circuit connection between two clients (UNI-C) of an optical network. This connection is achieved by signaling exchanges between UNI Client (UNI-C) and UNI Network (UNI-N) nodes. The UNI-C nodes are router nodes and UNI-N nodes are optical nodes.

- GMPLS UNI, on page 155
- GMPLS UNI Configurations, on page 162
- Verifying GMPLS UNI Configurations , on page 165
- Additional References, on page 174

GMPLS UNI

Prerequisites for Implementing GMPLS UNI

- An NCS 5500 Series router PID with CFP2 Digital Coherent Optics (DCO) form factor transceiver support.
- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- Router that runs Cisco IOS XR software.
- Installation of the Cisco IOS XR software mini-image on the router.
- Installation of the Cisco IOS XR MPLS software package on the router.
- Working knowledge of GMPLS. Refer the *Information About Implementing GMPLS UNI* section for details.

Restrictions for Implementing GMPLS UNI

- The total number of configured GMPLS UNI controllers should not exceed the platform scale limit of 500 GMPLS interfaces.
- Each UNI-N (ingress or egress) should be routable from its adjacent UNI-C. The UNI-C nodes need to be routable from the UNI-N nodes too.
- GMPLS UNI is supported only over DWDM controllers and so, over POS and GigabitEthernet interfaces.

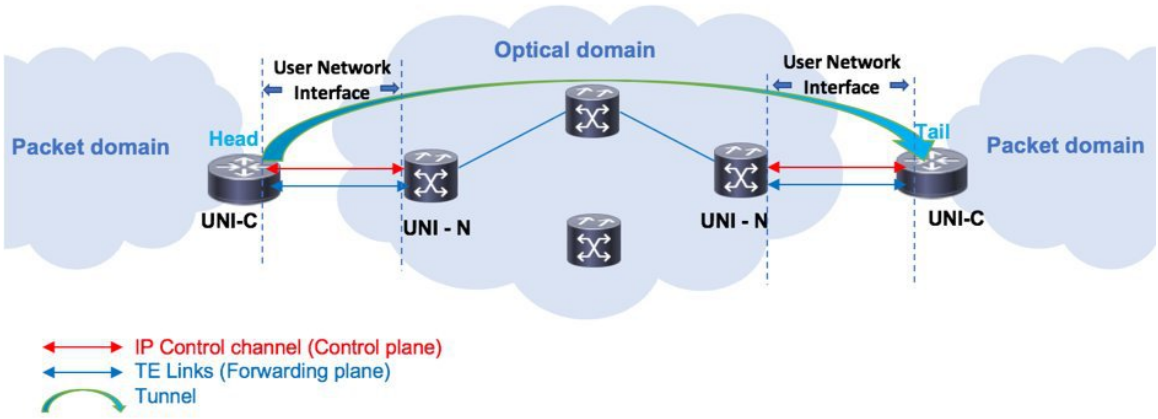
Information About Implementing GMPLS UNI

GMPLS UNI and GMPLS NNI

The GMPLS NNI optical network topology is known, and path calculations are performed at the NNI head. The GMPLS UNI optical network topology is unknown to the UNI-C nodes, and path calculations are performed by the UNI-N nodes.

GMPLS UNI Use Case

Figure 22: GMPLS UNI



The UNI components are UNI-N and UNI-C. The tunnel originates on the headend UNI, depicted in the left part of the image. The tunnel terminates on the tailend UNI, depicted in the right part of the image. Enable the following configurations on the headend UNI and tailend UNI:

- Control plane - IP control channel between the UNI-C and UNI-N router IDs. This creates LMP adjacency over the control channel.
- Forwarding plane - TE Link between UNI-C and UNI-N optical interfaces.
- Tunnel configuration from Head UNI-C to a Tail UNI-C optical interface, over the optical network.

  For each tunnel, you must enable corresponding tunnel and TE link configurations.

Link Management Protocol (LMP) – LMP manages the control channel across the UNIs, verifies TE link connectivity between the UNI interfaces, and performs fault management.

Dynamic LMP – You can enable the Dynamic LMP function which validates LMP configuration consistency across the headend and tailend UNIs. Consistency check examples:

- You have configured one end of a TE link as an unnumbered interface, and the other end with an IP address.
- You have entered the wrong neighbor interface ID when configuring an unnumbered neighbor interface.

Ensure that you enable the preceding configurations correctly.

GMPLS LSP Signaling

The GMPLS overlay model architecture is used for LSP signaling for GMPLS connections. In GMPLS UNI, UNI-C nodes send a request for a connection to UNI-N node. The connection request does not contain an end-to-end path. This is because, as mentioned previously, UNI-C nodes do not have knowledge of the topology of the optical network and therefore cannot determine the end-to-end path. The UNI-C node signals a connection request without an ERO.

The LSP diversity is signaled on a GMPLS UNI tunnel with a path-option. A path-option is permitted on a GMPLS UNI tunnel with a "no ERO" and an optional "XRO" attribute sets to specify LSP diversity requirements. If multiple LSP exclusions are configured in the attribute-set, they can be added to the path message along with an appropriate LSP connection diversity sub-object.

Supported LSP encoding types, and corresponding switching types:

Switching Type	LSP Encoding Type
LSC	Lambda
FSC	EthernetType1, EthernetType2, Fiber
DCSC	EthernetType2

A packet network is switched across a fiber, optic, or data channel network. Enable the LSP encoding and switching types under the GMPLS UNI and LMP configuration modes. Also, enable the Generalized PID (G-PID) under the GMPLS UNI configuration mode. G-PID is an identifier of the type of payload that the LSP carries, and the LSP endpoints (the UNI-C devices) use.

The LSP encoding type, switching type, and G-PID are updated to the GMPLS label.

Path Message without an ERO

In GMPLS UNI, UNI-C nodes send a request for a connection to UNI-N node. The connection request does not contain an end-to-end path, because, UNI-C nodes do not have knowledge of the topology of the optical network and therefore cannot determine the end-to-end path. The UNI-C node signals a connection request without an ERO.

When no ERO is present in a received path message, the UNI-N node calculates a route to the destination and includes that route in an ERO, before forwarding the path message. If no route is found, the UNI-N returns a path error message with an error code and subcode of *24,5 - "No route available toward destination"*.

The destination address of a GMPLS LSP can be either the optical router-id of the tail UNI-C node, or the optical address of the ingress interface to the tail UNI-C node. Supplying the router-id allows the UNI-N to route the tunnel to the tail UNI-C node via any attached UNI-N node; supplying the UNI-C's ingress interface address forces the tunnel's path to traverse the UNI-N node attached to that interface.



Note The optical router-ids and interface addresses may or may not be the same as the packet ones.

XRO Attribute-set

An optional XRO attribute-set can be specified as part of the path-option to specify LSP diversity requirements. An empty XRO attribute set results in the GMPLS tunnel being signaled with no exclusions, and therefore no XRO.



Note A non-existent XRO attribute-set can be configured in the GMPLS UNI tunnel path-option; in this case no attempt will be made to bring up the GMPLS tunnel until the configuration is complete.

Connection Diversity

Connection diversity is required to ensure that GMPLS tunnels can be established without sharing resources, thus, greatly reducing the probability of simultaneous connection failures. For example, an edge-node wishes to establish multiple LSPs towards the same destination edge-node, and these LSPs need to have few or no resources in common.

Connection diversity supports the establishment of a GMPLS LSP which is diverse from the path taken by an existing LSP. An XRO is added to the tunnel's path message with appropriate LSP diversity sub-objects or exclusions. A maximum of 20 connection diversity exclusions per XRO is supported.

GMPLS RSVP VRF Signaling

The Cisco IOS XR software supports a single non-default VRF for the GMPLS RSVP signaling. This allows GMPLS signaling to work even when the only available communication between the UNI-C and UNI-N nodes is through a VRF. This non-default VRF is supported only for GMPLS signaling; whereas the MPLS-TE signaling continues to support only the default VRF.

DWDM Transponder Integration

A GMPLS UNI based solution preserves all the advantages of the integration of the DWDM transponder into the router blade. These advantages include the following:

- Improved CAPEX and OPEX models.
- Component, space and power savings.
- Improved IP availability through pro-active protection.

nLight Enhancements

These topics describe the enhancements made to nLight (also known as GMPLS UNI):

Explicit Route Object

Explicit Route Objects (EROs) limit LSP routing to a specified list of LSRs. Formerly, the UNI Client (UNI-C) node signaled a connection request, without an ERO, to the UNI Network (UNI-N) node. In this IOS XR Software release, the UNI-C node provides support for path message with ERO for GMPLS tunnels. This includes the capability to specify either a strict or a loose ERO to a path option to be included in the path message for processing by the ingress UNI-N.

An ERO in constructed using the strict and loose hops, specified in the explicit path, by the path option.

When a loose hop is configured, it identifies one or more transit LSRs which suggests the preferred path for the LSP. If a suggested path fails, another LSR is tried.

When a strict hop is configured, it identifies an exact path through which the LSP must be routed. Strict hop EROs specify the exact sequence of LSRs in the LSP.

As a result of these operations, a LSP is established from the sender to the destination of the session, following the explicitly routed path specified in the ERO.



Note

- *lockdown* and *verbatim* are mandatory in ERO path option.
- A path option may still be configured to use no ERO.

  In no ERO, *lockdown* is mandatory.

Wavelength Specification

The wavelength (also called label) specification enhancement enables the network planning tool to determine the wavelength, and specify the same at the UNI-C. The UNI-N then accepts the label provided by the UNI-C, or rejects the path entirely. Previously, the wavelength to be used for the GMPLS UNI tunnel was determined by the UNI-N, taking into account the headend UNI-C's capabilities.

The wavelength to be used is added to the path option configuration. This optional configuration allows a fixed wavelength to be specified for the path option.

When signaling using a path option with the specified wavelength takes place, the following changes happen because of the wavelength specification enhancement:

- The configured wavelength is validated against the controller’s capabilities; signaling fails if the wavelength cannot be used by the controller.
- The upstream label is set to the specified wavelength.
- The label-set in the Path message, instead of containing one label for each supported wavelength, contains only the specified wavelength.
- A path-error message with error code 25 and subcode 6 no longer receives special handling. If a suggested label is supplied, it is ignored.



Note A suggested label received in response to signaling with a path option that specifies a different label, is not stored for future use. Other path options, in general, have different constraints and therefore require path calculation to be redone.

Multiple Path Options

Multiple path options are permitted per GMPLS UNI tunnel. The index given to each path option indicates its relative preference level, with lower indices being preferred. This is similar to the existing multiple path option functionality available for packet TE. This allows the provision of multiple path options with, for example, progressively free constraints.

The path-option index is no longer fixed to ten and is now set by the user and distinguishes path options in the same manner as for packet tunnels. In all situations where a tunnel is being brought up or reoptimized, the path-option with the lowest index is tried first; if no LSP can be established with this path option, then

subsequent path options are tried in ascending order. This also applies to recovery from failures, unless any recovery path option is specified.

Reoptimization

Reoptimization differs from restoration though the mechanisms involved are similar. Reoptimization occurs without the original connection having failed.

Unlike packet tunnels, reoptimization in GMPLS tunnels is not supposed to be loss free.

Manual Reoptimization

Manual reoptimization of a single GMPLS UNI tunnel can be triggered from the UNI-C node (headend). Use the **mpls traffic-eng optical-uni reoptimize tunnel-id** command to trigger manual reoptimization of a GMPLS UNI tunnel.

The manual trigger for reoptimization causes the currently established LSP to be torn down and signals a new LSP using the normal bring-up process (though the new LSP is same as the current one).

It is not possible to trigger reoptimization for multiple GMPLS UNI tunnels or at the tailend of a tunnel.

SRLG Discovery



Note Shared Risk Link Group (SRLG) discovery, SRLG collection and SRLG recording represent the same function.

The head and tail UNI-C routers have no direct knowledge of the path taken through the optical network by a GMPLS UNI tunnel, or of the properties of that path. All information about the path of a particular GMPLS UNI connection must therefore be explicitly requested and learned during the signaling process.

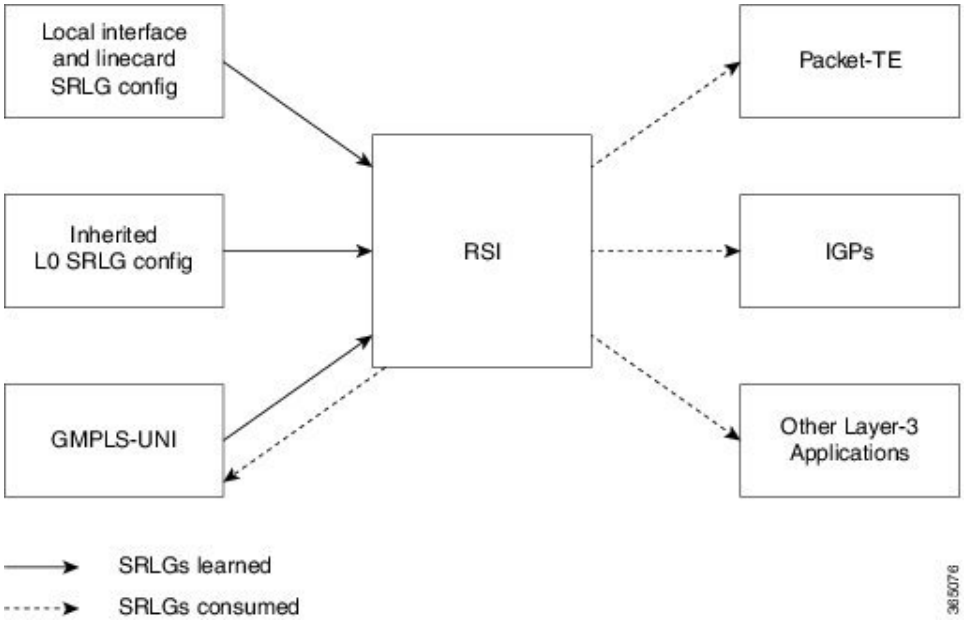
A key property of a GMPLS UNI connection is the set of SRLGs used by the optical links along the connection. It is necessary for the UNI-C routers to learn the set of SRLGs associated with a connection, so that this information can be used, both by GMPLS UNI in the specification of diversity requirements for other connections and by Layer-3 applications for effecting routing and protection decision making.

The learning of SRLGs during GMPLS UNI LSP signaling is done by requesting SRLG collection when LSP signaling is initiated, and by the addition of SRLG RRO sub-objects to the Path and Resv messages during signaling as described in IETF draft *SRLG-collect*. Path message learns egress interfaces from head to tail and Resv message learns egress interfaces from tail to head.

Provision of Discovered SRLGs to RSI

Once the SRLGs used by a GMPLS UNI connection are collected during signaling as in SRLG discovery, they are made available to the Layer-3 processes. This is done through RSI (Router Space Infrastructure), as illustrated in the following diagram:

Figure 23: SRLG Communication



An API is provided by the RSI component to allow SRLGs discovered during GMPLS UNI signaling to be communicated to RSI, as documented in IETF draft *RSI-SRLG*. RSI combines the SRLG sets learned from GMPLS and configuration for an interface and deliver a single set of SRLGs to applications registered as SRLG clients.

The SRLGs discovered during GMPLS UNI signaling are given to RSI for application to the Layer-3 interface of the DWDM controller associated with the GMPLS UNI tunnel. This may be a POS, GigE or an OTN interface.

SRLG Announce

All SRLGs discovered through GMPLS signaling are announced to RSI once the tunnel is up. These SRLGs are withdrawn from RSI when the tunnel goes down.

SRLG Diversity

Note SRLG diversity and SRLG exclusion represent the same function.

Support is added for signaling SRLG based diversity requirements, based on the XRO SRLG sub-object defined in RFC 4874. The use of SRLGs removes the restrictions of LSP based diversity, as SRLGs are flooded throughout the optical network, and by their very nature, reduce the risk of concurrent failure.

SRLG diversity is configured under the XRO attribute-set.

Head UNI-C Behavior

SRLG diversity is configured at the tunnel head. Individual SRLG exclusions are added to an XRO attribute-set; each is specified as either *best-effort* or mandatory (*strict*). Whenever any exclusion is specified, an XRO object is added to the Path message by the head UNI-C. The XRO contains a SRLG sub-object for each specified SRLG. The SRLG exclusions may coexist in the same XRO with LSP exclusions.

The XRO attribute-set is associated with tunnel path options in the same manner as for LSP exclusions.

If a SRLG with a strict exclusion matches an SRLG configured on the local DWDM controller, the bring-up attempt fails.

The SRLG exclusions requested by the head UNI-C are processed by the ingress UNI-N node during path calculation for the tunnel.

Tail UNI-C Behavior

On receiving a Path message containing an XRO, the tail UNI-C inspects each SRLG sub-object. If a SRLG sub-object, with a strict exclusion, matches an SRLG configured on the local DWDM controller, the Path message is rejected and a path-error is generated with error codes. No action is taken if the SRLG sub-object specifies a *best-effort* exclusion.

Multi-Layer Restoration - Optical

Multi-Layer Restoration-Optical (MLR-O) involves restoration from failures in the optical network that can leverage the same router interfaces at both ends.

Optical restoration involves the repair of a failure by the optical network locally. Although the routers may see loss of light until the failure is repaired, there is no signaling involving the routers, and from the routers perspective the GMPLS UNI LSP remains unchanged.

Optical Restoration: Same Wavelength

When a failure occurs on a physical link within the optical network, the routers identify that the link is down and Layer 3 protection mechanisms, such as FRR, are used to minimize the traffic loss. The optical network re-routes the GMPLS connection to an alternative path. This is done without any involvement of the routers.

Limitation - A significant limitation of optical restoration in this case, is that the wavelength in use for the connection cannot be changed. This is because the wavelength must be the same along the entire path and cannot be changed without end-to-end signaling. The constraints imposed on the connection during its initial signaling are also unchanged, which may reduce the chance of finding an alternative path.

Optical Restoration: Wavelength Change

Optical restoration may occur with an associated wavelength change, in the case where the optical network finds an alternative path with the same constraints as were originally signaled, but using a different wavelength. Some signaling is required, since the wavelength (and therefore the labels) used by the GMPLS connection are to change.

Consider a failure within the optical network on the path of a GMPLS UNI LSP. The restoration proceeds as in the previous case (same wavelength), but the new path found, uses a different wavelength. The ingress UNI-N then sends a path-error message indicating the new wavelength to be used; this has error code 24 (routing), sub-error 6 (unacceptable label set) and contains a suggested-label sub-object with the new label to be used. The head UNI-C then signals a new LSP with the new wavelength.

Although the wavelength in use may change in this case, the constraints used in signaling the original LSP remain unchanged.

GMPLS UNI Configurations

Configuring TE for GMPLS UNI

TE configuration specific to packet tunnels does not affect GMPLS UNI tunnels.

Configuring GMPLS UNI Controller

Perform this task to setup a GMPLS tail in MPLS-TE configuration. This task enables GMPLS UNI controller submode to configure controllers for establishing GMPLS UNI tunnels. This is the minimal configuration required at the tunnel tail.

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# gmpls optical-uni
Router(config-te-gmpls)# controller dwdm 0/1/0/1
```

Configuring the GMPLS UNI Controller as a Tunnel Head

Perform this task to configure the tunnel properties for a GMPLS UNI controller.

This configuration designates the controller as a tunnel-head, rather than a tunnel tail. After you configure the tunnel properties, the incoming path messages are rejected and any existing tailend tunnel is torn down.

```
Router(config-te-gmpls-cntl)# mtu 9000
Router(config-te-gmpls-cntl)# tunnel-properties
Router(config-te-gmpls-tun)# g-pid 37
Router(config-te-gmpls-tun)# encoding-type lambda
Router(config-te-gmpls-tun)# tunnel-id 100
Router(config-te-gmpls-tun)# logging events lsp-status state
Router(config-te-gmpls-tun)# destination ipv4 unicast 10.10.3.4
Router(config-te-gmpls-tun)# exit
```

This configuration assigns the Lambda switching type for transporting traffic.

```
Router(config-te-gmpls-cntl)# switching-type lsc
```

(Optional) Tunnel Properties

```
Router(config-te-gmpls-cntl)# tunnel-properties
Router(config-te-gmpls-tun)# priority 3 2
Router(config-te-gmpls-tun)# record-route
Router(config-te-gmpls-tun)# signalled-name sign1
Router(config-te-gmpls-tun)# exit
```

Note If you remove the GMPLS UNI and GMPLS UNI controller submodes, all GMPLS UNI tunnels are removed.

Path Options - These configurations specify a path-option for the GMPLS tunnel headend router.

This configures a path option with no ERO.

```
Router(config-te-gmpls-tun)# path-option 30 no-ero lockdown
```

This configures a path option with an ERO for the GMPLS tunnel. You can specify an explicit path in the range 1 to 1000. **lockdown** and **verbatim** are mandatory in an ERO path option.

```
Router(config-te-gmpls-tun)# path-option 10 explicit name explicit_path_a lockdown verbatim
```

This configures a wavelength for the path option. The DWDM channel number range is 1 to 89. The DWDM channel number configured is formulated as 61-channel number. For example, for channel number 42 (in the supported channel list), the configured "DWDM channel number" is 61 - 42 = 19.

```
Router(config-te-gmpls-tun)# path-option 10 explicit name exp_all_loose_hop signaled-label
 dwdm wavelength 10 lockdown verbatim
```

This configures multiple path options for a single tunnel.

```
Router(config-te-gmpls-tun)# path-option 10 explicit name explicit_path_a lockdown verbatim
```

```
Router(config-te-gmpls-tun)# path-option 20 explicit name explicit_path_b lockdown verbatim
Router(config-te-gmpls-tun)# path-option 30 no-ero lockdown
```

This configures the LSP diversity function. You can configure an XRO attribute-set as part of the path-option for MPLS-TE, and specify exclusions for an attribute set for LSP diversity.

```
Router(config-te-gmpls-tun)# path-option 10 no-ero xro-attribute-set A01 lockdown
```

This configures the connection diversity function. You can specify exclusions for an attribute set for LSP diversity.

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# attribute-set xro attrset01
Router(config-te-attribute-set)# exclude best-effort lsp source 10.10.1.2 destination
10.20.4.4 tunnel-id 17 extended-tunnel-id 10.20.3.3 lsp-id 17
Router(config-te-attribute-set)# commit
```

Configuring LMP for GMPLS UNI

To implement LMP configuration for GMPLS UNI, follow these steps:

LMP Configuration - Configure the optical router ID and LMP neighbor:

```
Router# configure
Router(config)# lmp
Router(config-lmp)# gmpls optical-uni
Router(config-lmp-gmpls-uni)# router-id ipv4 unicast 10.10.4.4
Router(config-lmp-gmpls-uni)# neighbor nbr1
Router(config-lmp-gmpls-uni-nbr1)# dynamic
Router(config-lmp-gmpls-uni-nbr1)# hello 1000 10000
Router(config-lmp-gmpls-uni-nbr-nbr1)# ipcc routed
Router(config-lmp-gmpls-uni-nbr-nbr1)# router-id ipv4 unicast 10.10.4.5
```

LMP Controller Configuration - Configure the LMP link for the GMPLS UNI controller:

```
Router(config-lmp-gmpls-uni)# controller dwdm 0/4/0/0
Router(config-lmp-gmpls-uni-cntl)# neighbor nbr1
Router(config-lmp-gmpls-uni-cntl)# link-id ipv4 unicast 10.2.2.4
Router(config-lmp-gmpls-uni-cntl)# switching-type lsc
Router(config-lmp-gmpls-uni-cntl)# encoding-type Lambda
Router(config-lmp-gmpls-uni-cntl)# neighbor link-id ipv4 unicast 10.2.2.5
Router(config-lmp-gmpls-uni-cntl)# neighbor interface-id unnumbered 17
Router(config-lmp-gmpls-uni-cntl)# commit
```

Configuring RSVP Optical Refresh Interval and Missed Count

Perform this task to configure optical refresh interval under the RSVP controller submode and to configure the number of missed refresh messages allowed before optical tunnel states are deleted.

```
Router# configure
Router(config)# rsvp
Router(config-rsvp)# controller dwdm 0/1/0/1
Router(config-rsvp-cntl)# signalling refresh out-of-band interval 200
Router(config-rsvp-cntl)# signalling refresh out-of-band missed 30
Router(config-rsvp-cntl)# commit
```

Configuring SRLG

SRLG discovery on the head router of a nLight tunnel.

- SRLG discovery/recording is enabled only on the headend for each tunnel.

- SRLG discovery/recording allows a maximum of 62 SRLGs in RSVP, which is different from the maximum count of 64 in RSI.

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# gmpls optical-uni
Router(config-te-gmpls)# controller dwdm 0/1/0/1
Router(config-te-gmpls-cntl)# tunnel-properties
Router(config-te-gmpls-tun)# logging events lsp-status state
Router(config-te-gmpls-tun)# tunnel-id 100
Router(config-te-gmpls-tun)# record srlg
Router(config-te-gmpls-tun)# destination ipv4 unicast 10.10.3.4
Router(config-te-gmpls-tun)# path-option 10 no-ero lockdown
Router(config-te-gmpls-tun)# exit
```

To announce the discovered SRLGs to the system, use the **announce srlgs** command.

```
Router(config-te-gmpls-cntl)# announce srlgs
Router(config-te-gmpls-cntl)# commit
```

You can configure the attribute-set and SRLG path diversity with the best-effort or strict exclusion function.

```
Router# configure
Router(config)# mpls traffic-eng
Router(config-mpls-te)# attribute-set xro exclude_srlgs
Router(config-te-attribute-set)# exclude best-effort srlg value 21
Router(config-te-attribute-set)# exit
```

Associate the XRO attribute set to the GMPLS UNI tunnel through the path option. The path-option range is 1 to 1000.

```
Router(config-mpls-te)# gmpls optical-uni
Router(config-te-gmpls)# controller dwdm 0/1/0/1
router(config-te-gmpls-cntl)# tunnel-properties
Router(config-te-gmpls-tun)# path-option 10 no-ero xro-attribute-set exclude_srlgs lockdown
```

Verifying GMPLS UNI Configurations

Various GMPLS UNI function configurations are displayed in this section:

Explicit Route Object (ERO)

```
Router# show mpls traffic-eng tunnels 1001 detail

Name: GMPLS-UNI-dwdm0_3_0_0 Destination: 172.16.0.1 Signalled-Name: head_ot1001_172.16.0.1

GMPLS UNI tunnel controlling link dwdm0/3/0/0, tunnel-id: 1001
Status:
Admin:    up Oper:   up   Path:  valid   Signalling: connected
path option 10, (LOCKDOWN verbatim) type explicit explicit_path_a (Basis for Setup, path
weight 0)
G-PID: 0x0800 (derived from egress interface properties) Creation Time: Fri Jul 17 08:41:21
 ---- (3d07h ago)
..
Current LSP Info:
  Instance: 20
  Uptime: 00:00:33 (since Mon Jul 20 ---- 15:45:22)

Upstream label:
```

```
  Optical label:
  Grid              : DWDM
  Channel spacing   : 50GHz
  Identifier        : 0
  Channel Number    : 60

Downstream label:

  Optical label:
  Grid              : DWDM
  Channel spacing   : 50GHz
  Identifier        : 0
  Channel Number    : 60

Router-IDs: local   10.0.0.1
downstream          172.16.0.1
Soft Preemption: None
SRLGs: not collected
Path Info:
  Outgoing:
    Explicit Route:
        Strict, 10.10.10.2
        Strict, 10.11.11.3
        Strict, 10.12.12.3
```

Wavelength

The following sequence of examples show how to add a wavelength to a path option for a GMPLS tunnel and verify the outgoing label is set accordingly.

This example shows how to configure a GMPLS tunnel with no ERO path option.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
    path-option 10 no-ero lockdown
   !
```

This example shows how to verify the default values for the outgoing label (UNI-N source, channel number same as Default Channel) and the list of valid wavelengths.

```
Router# show mpls traffic-eng link-management optical-uni controller dwdm 0/3/0/0

Optical interface: dwdm0/3/0/0
  Overview:
    IM state: Up
    Child interface: POS0_3_0_0: IM state Up
    OLM/LMP state: Up
    Optical tunnel state: up
  Connection:
    Tunnel role: Head
    Tunnel-id: 1001, LSP-id 21, Extended tunnel-id 10.0.0.1
    Tunnel source: 10.0.0.1, destination: 172.16.0.1
    Optical router-ids: Local: 10.0.0.1, Remote: 172.16.0.1
    Label source: UNI-N
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 60
    Downstream label:
```

```
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 60
    SRLG discovery: Disabled
    SRLG announcement: None
...
  Optical capabilities:
    Controller type: DWDM
    Channel spacing: 50 GHz
    Default channel: 60
    89 supported channels:
      -28, -27, -26, -25, -24, -23, -22, -21
      -20, -19, -18, -17, -16, -15, -14, -13
      -12, -11, -10, -9, -8, -7, -6, -5
      -4, -3, -2, -1, 0, 1, 2, 3
      4, 5, 6, 7, 8, 9, 10, 11
      12, 13, 14, 15, 16, 17, 18, 19
      20, 21, 22, 23, 24, 25, 26, 27
      28, 29, 30, 31, 32, 33, 34, 35
      36, 37, 38, 39, 40, 41, 42, 43
      44, 45, 46, 47, 48, 49, 50, 51
      52, 53, 54, 55, 56, 57, 58, 59
      60
    Controller SRLGs
      None
```

This example shows how to set valid wavelength 10 (61 - 51) for the current path.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
   path-option 10 explicit name explicit_all_loose_multi_hop signaled-label dwdm wavelength
 10 lockdown verbatim
   !
```

This example shows how to verify that the tunnel is up and the specified wavelength is used (label source is UNI-C and outgoing label is 51).

```
Router# show mpls traffic-eng link-management optical-uni controller dwdm 0/3/0/0

Optical interface: dwdm0/3/0/0
  Overview:
    IM state: Up
    Child interface: POS0_3_0_0: IM state Up
    OLM/LMP state: Up
    Optical tunnel state: up
  Connection:
    Tunnel role: Head
    Tunnel-id: 1001, LSP-id 23, Extended tunnel-id 10.0.0.1
    Tunnel source: 10.0.0.1, destination: 172.16.0.1
    Optical router-ids: Local: 10.0.0.1, Remote: 172.16.0.1
    Label source: UNI-C
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    Downstream label:
      Optical label:
```

```
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    SRLG discovery: Disabled
    SRLG announcement: None
...
  Optical capabilities:
    Controller type: DWDM
    Channel spacing: 50 GHz
    Default channel: 60
    89 supported channels:
      -28, -27, -26, -25, -24, -23, -22, -21
      -20, -19, -18, -17, -16, -15, -14, -13
      -12, -11, -10, -9, -8, -7, -6, -5
      -4, -3, -2, -1, 0, 1, 2, 3
      4, 5, 6, 7, 8, 9, 10, 11
      12, 13, 14, 15, 16, 17, 18, 19
      20, 21, 22, 23, 24, 25, 26, 27
      28, 29, 30, 31, 32, 33, 34, 35
      36, 37, 38, 39, 40, 41, 42, 43
      44, 45, 46, 47, 48, 49, 50, 51
      52, 53, 54, 55, 56, 57, 58, 59
      60
    Controller SRLGs
      None
```

This example shows how to verify the upstream label on the tunnel tail.

```
Router# show mpls traffic-eng link-management optical-uni controller dwdm 0/3/0/0

Optical interface: dwdm0/3/0/0
  Overview:
    IM state: Up
    Child interface: POS0_3_0_0: IM state Up
    OLM/LMP state: Up
    Optical tunnel state: up
  Connection:
    Tunnel role: Tail
    Tunnel-id: 1001, LSP-id 23, Extended tunnel-id 10.0.0.1
    Tunnel source: 10.0.0.1, destination: 172.16.0.1
    Optical router-ids: Local: 172.16.0.1, Remote: 10.0.0.1
    Label source: UNI-N
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    Downstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 51
    SRLG discovery: Disabled
    SRLG announcement: None
…..
  Optical capabilities:
    Controller type: DWDM
    Channel spacing: 50 GHz
    Default channel: 60
    89 supported channels:
      -28, -27, -26, -25, -24, -23, -22, -21
      -20, -19, -18, -17, -16, -15, -14, -13
```

```
      -12, -11, -10, -9, -8, -7, -6, -5
      -4, -3, -2, -1, 0, 1, 2, 3
      4, 5, 6, 7, 8, 9, 10, 11
      12, 13, 14, 15, 16, 17, 18, 19
      20, 21, 22, 23, 24, 25, 26, 27
      28, 29, 30, 31, 32, 33, 34, 35
      36, 37, 38, 39, 40, 41, 42, 43
      44, 45, 46, 47, 48, 49, 50, 51
      52, 53, 54, 55, 56, 57, 58, 59
      60
    Controller SRLGs
      None
```

Multiple Path Options

This example shows how to configure multiple path options.

```
mpls traffic-eng
 gmpls optical-uni
   controller dwdm0/2/0/2
    tunnel-properties
     path-option 10 explicit name explicit_path_a lockdown verbatim
     path-option 20 explicit name explicit_path_b lockdown verbatim
     path-option 30 no-ero lockdown
    !
```

The following sequence of examples show how to configure a GMPLS tunnel, add a new path option with a lower index than the path option in use, flap the tunnel and verify that the new path option (with a lower index) is used.

This example shows how to configure a GMPLS tunnel with one path option.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
    path-option 10 explicit name explicit_path_a lockdown verbatim
   !
```

This example shows how to verify the tunnel path and status with a show command.

```
Router# show mpls traffic-eng tunnels 1001 detail

Name: GMPLS-UNI-dwdm0_3_0_0  Destination: 172.16.0.1
  Signalled-Name: head_ot1001_172.16.0.1
GMPLS UNI tunnel controlling link dwdm0/3/0/0, tunnel-id: 1001
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (LOCKDOWN verbatim) type explicit explicit_path_a (Basis for Setup,
path weight 0)
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Fri Jul 17 08:41:21 ---- (3d06h ago)
..
```

This example shows how to add another path option with a lower index.

```
gmpls optical-uni
  controller dwdm0/3/0/0
   tunnel-properties
    tunnel-id 1001
    destination ipv4 unicast 172.16.0.1
    path-option 1 no-ero lockdown
```

```
    path-option 10 explicit name explicit_path_a lockdown verbatim
   !
```

Flag the tunnel (or trigger reoptimization) and verify that the tunnel comes up on the path with a lower index.

```
Router# show mpls traffic-eng tunnels 1001 detail

Name: GMPLS-UNI-dwdm0_3_0_0  Destination: 172.16.0.1
  Signalled-Name: head_ot1001_172.16.0.1
GMPLS UNI tunnel controlling link dwdm0/3/0/0, tunnel-id: 1001
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 1, (LOCKDOWN) type no-ero  (Basis for Setup, path weight 0)
    Last Signalled Error : Mon Jul 20 17:03:00 2015
      Info: [24] PathErr(2,2)-(Admin, reason unknown) at 50.0.0.2
    path option 10, (LOCKDOWN verbatim) type explicit explicit_all_loose_multi_hop
    Last Signalled Error : Mon Jul 20 17:03:00 ----
      Info: [25] PathErr(2,2)-(Admin, reason unknown) at 50.0.0.2
```

SRLG

SRLG Discovery

```
Router# show mpls traffic-eng tunnels 100 detail

Name: GMPLS-UNI-dwdm0_2_0_0  Destination: 192.168.1.2
  Signalled-Name: rtrA_ot100_192.168.1.2
GMPLS UNI tunnel controlling link dwdm0/2/0/0, tunnel-id: 100
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (LOCKDOWN) type no-ero  (Basis for Setup, path weight 0)
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Mon Jul 20 19:32:03 ---- (00:48:02 ago)
  Config Parameters:
    Priority:  7  7 Affinity: 0x0/0xffff
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    SRLG discovery: Enabled
….
 Soft Preemption: None
    SRLGs: mandatory collection
    Path Info:
  ….
    Resv Info:
      Record Route:
        IPv4 10.10.10.2, flags 0x0
        SRLGs: 21, 22, 23, 24
      Fspec: avg rate=10000 kbits, burst=1000 bytes, peak rate=10000 kbits
Displayed 1 (of 3) heads, 0 (of 0) midpoints, 0 (of 2) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

This example shows how to verify SRLG discovery configuration at the headend and the tailend. The output shows the list of SRLGs.

```
Router# show srlg

  System Information::
  Interface Count          : 4 (Maximum Interfaces Supported 512)
  Group Count              : 0 (Maximum Groups Supported 50)
  Inherit Location Count   : 0 (Maximum Inherit Locations Supported 10)
  Optical Interfaces Count : 4 (Maximum Optical Interfaces Supported 500)
```

```
  Interface    : GigabitEthernet0/2/0/0, Value Count : 10, Registrations : 2
  SRLG Values  : 11, 12, 13, 14, 15
                 21, 22, 23, 24, 25                                          -> Note:
These are announced srlgs.

  Interface    : GigabitEthernet0/2/0/2, Value Count : 0, Registrations : 1
  SRLG Values  :

  Interface    : GigabitEthernet0/2/0/4, Value Count : 0, Registrations : 2
  SRLG Values  :

  Interface    : GigabitEthernet0/2/0/5, Value Count : 0, Registrations : 2
  SRLG Values  :

  Optical Interface: dwdm0/2/0/0, Value Count : 5, References: 2
  SRLG Values      : 11, 12, 13, 14, 15                                       -> Note:
These are locally configured srlgs for controller (dwdm)

  Optical Interface: dwdm0/2/0/1, Value Count : 0, References: 1
  SRLG Values      :

  Optical Interface: dwdm0/2/0/2, Value Count : 0, References: 1
  SRLG Values      :

  Optical Interface: dwdm0/2/0/3, Value Count : 0, References: 1
  SRLG Values      :
```

SRLG Announce

```
Router# show srlg

  System Information::
  Interface Count          : 2 (Maximum Interfaces Supported 512)
  Group Count              : 0 (Maximum Groups Supported 50)
  Inherit Location Count   : 0 (Maximum Inherit Locations Supported 10)
  Optical Interfaces Count : 5 (Maximum Optical Interfaces Supported 500)

  Interface    : GigabitEthernet0/2/0/4, Value Count : 0, Registrations : 2
  SRLG Values  :

  Interface    : GigabitEthernet0/2/0/5, Value Count : 0, Registrations : 2
  SRLG Values  :

  Interface: GigabitEthernet0/2/0/0, Value Count : 4, References: 1
  SRLG Values      : 21, 22, 23, 24

  Optical Interface: dwdm0/2/0/0, Value Count : 3, References: 2
  SRLG Values      : 11, 12, 13

  Optical Interface: dwdm0/2/0/1, Value Count : 0, References: 1
  SRLG Values      :

  Optical Interface: dwdm0/2/0/2, Value Count : 0, References: 1
  SRLG Values      :

  Optical Interface: dwdm0/2/0/3, Value Count : 0, References: 1
  SRLG Values      :
```

SRLG Diversity

```
Router# show mpls traffic-eng tunnels 100 detail

Name: GMPLS-UNI-dwdm0_2_0_0  Destination: 192.168.1.2
  Signalled-Name: rtrA_ot100_192.168.1.2
GMPLS UNI tunnel controlling link dwdm0/2/0/0, tunnel-id: 100
```

```
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (LOCKDOWN) type no-ero  (Basis for Setup, path weight 0)
      XRO attribute-set: exclude_srlgs
        Best-effort, SRLG id 21
    Last Signalled Error : Mon Jul 20 20:55:33 ----
      Info: [5] PathErr(24,67)-(routing, route blocked by exclude route) at 10.10.10.2
    G-PID: 0x0800 (derived from egress interface properties)
    Creation Time: Mon Jul 20 19:32:03 2015 (01:25:19 ago)
  Config Parameters:
    Priority:  7  7 Affinity: 0x0/0xffff
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    SRLG discovery: Enabled
  Binding Label: 0
  History:
    Tunnel has been up for: 00:00:23 (since Mon Jul 20 20:56:59 EDT 2015)
    Current LSP:
      Uptime: 00:00:23 (since Mon Jul 20 20:56:59 EDT ----)
  Current LSP Info:
    Instance: 6
    Uptime: 00:00:23 (since Mon Jul 20 20:56:59 EDT ----)
    Upstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 16
    Downstream label:
      Optical label:
      Grid                : DWDM
      Channel spacing     : 50 GHz
      Identifier          : 0
      Channel Number      : 16
    Router-IDs: local      192.168.1.1
                downstream 192.168.1.2
    Soft Preemption: None
    SRLGs: mandatory collection
    Path Info:
      Outgoing:
        No ERO
      Route Exclusions:
        Best-effort, SRLG id 21
      Record Route: Disabled
      Tspec: avg rate=10000 kbits, burst=1000 bytes, peak rate=10000 kbits
      Session Attributes: Local Prot: Not Set, Node Prot: Not Set, BW Prot: Not Set
    Resv Info:
      Record Route:
        IPv4 10.10.10.2, flags 0x0
        SRLGs: 21, 22, 23, 24
      Fspec: avg rate=10000 kbits, burst=1000 bytes, peak rate=10000 kbits
Displayed 1 (of 3) heads, 0 (of 0) midpoints, 0 (of 2) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

GMPLS UNI Running Configuration

Head UNI-C for a GMPLS Tunnel

```
rsvp
    controller dwdm 0/1/0/1
      signalling refresh out-of-band interval 3600
      signalling refresh out-of-band missed 24
```

```
  !
!
mpls traffic-eng
  gmpls optical-uni
    controller dwdm 0/1/0/1
      tunnel-properties
        tunnel-id 100
        destination 100.20.20.20
        path-option 10 no-ero
      !
    !
  !
!
lmp
  gmpls optical-uni
    router-id 100.11.11.11
    neighbor nbr_A
      ipcc routed
      neighbor router-id ipv4 unicast 100.12.12.12
    !
    controller dwdm 0/1/0/1
      neighbor nbr_A
      link-id ipv4 unicast 192.168.100.1
      neighbor link-id ipv4 unicast 192.168.100.2
      neighbor interface-id unnumbered 13
    !
```

Tail UNI-C for a GMPLS Tunnel

Note The controller must be specified under the GMPLS UNI submode to inform TE that incoming GMPLS path messages are to be accepted and processed.

```
rsvp
    controller dwdm 0/1/0/1
      signalling refresh out-of-band interval 3600
      signalling refresh out-of-band missed 24
    !
  !
  mpls traffic-eng
    gmpls optical-uni
      controller dwdm 0/1/0/1
      !
    !
  !
  lmp
    gmpls optical-uni
      router-id 100.20.20.20
      neighbor nbr_B
        ipcc routed
        neighbor router-id ipv4 unicast 100.19.19.19
      !
      controller dwdm 0/1/0/1
        neighbor nbr_B
        link-id ipv4 unicast 192.168.103.2
        neighbor link-id ipv4 unicast 192.168.103.1
        neighbor interface-id unnumbered 22
      !
```

LSP Diversity

This example shows the configuration for two diverse LSPs:

```
mpls traffic-eng
   attribute-set xro exclude-tun1
    exclude best-effort lsp source 88.0.0.8 destination 10.0.0.2 tunnel-id 1
extended-tunnel-id 88.0.0.8
    !
   attribute-set xro exclude-tun2
    exclude strict lsp source 88.0.0.8 destination 10.0.1.2 tunnel-id 2 extended-tunnel-id
 88.0.0.8 lsp-id 2
    !
   gmpls optical-uni
    controller dwdm 0/1/0/0
     tunnel-properties
      logging events lsp-status state
      tunnel-id 1
      destination ipv4 unicast 10.0.0.2
      path-option 10 no-ero xro-attribute-set exclude-tun2
     !
    !
    controller dwdm 0/1/0/1
     tunnel-properties
      logging events lsp-status state
      tunnel-id 2
      destination ipv4 unicast 10.0.1.2
      path-option 10 no-ero xro-attribute-set exclude-tun1
     !
    !
```

Additional References

Standards

Standard	Title
OIF UNI 1.0	*User Network Interface (UNI) 1.0 Signaling Specification*

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

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 3471	*Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description*
RFC 3473	*Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions*
RFC 4208	*Generalized Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model*
RFC 4872	*RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery*
RFC 4874	*Exclude Routes - Extension to Resource ReserVation Protocol-Traffic Engineering (RSVP-TE)*
RFC 6205	*Generalized Labels for Lambda-Switch-Capable (LSC) Label Switching Routers*

Technical Assistance

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