MPLS Configuration Guide for Cisco NCS 6000 Series Routers, Cisco IOS-XR Release 6.3.x

First Published: 2018-03-01

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

CONTENTS

Preface

Note This product has reached end-of-life status. For more information, see the End-of-Life and End-of-Sale Notices.

The preface contains these sections:

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

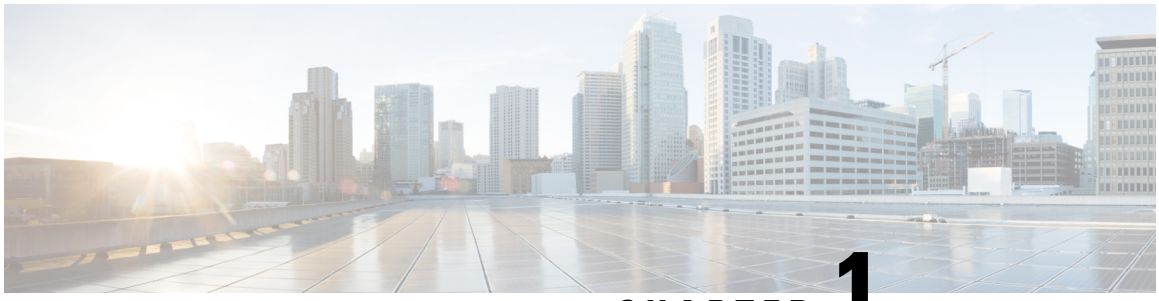
Date	Change Summary
September 2017	Initial release of this document.
March 2018	Republished for 6.3.2.

Communications, Services, and Additional Information

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

Cisco Bug Search Tool

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



CHAPTER 1

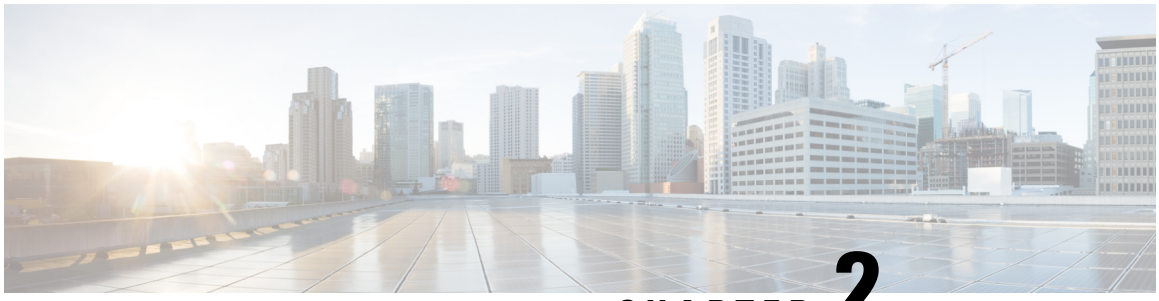
New and Changed MPLS Features

This table summarizes the new and changed feature information for the *MPLS Configuration Guide for Cisco NCS 6000 Series Routers*, and tells you where they are documented.

New and Changed MPLS Feature Information

Table 2: New and Changed Features

Feature	Description	Changed in Release	Where Documented
None	No new features introduced	Not applicable	Not applicable



CHAPTER 2

Implementing MPLS Label Distribution Protocol

The Multiprotocol Label Switching (MPLS) is a standards-based solution driven by the Internet Engineering Task Force (IETF) that was devised to convert the Internet and IP backbones from best-effort networks into business-class transport mediums.

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 ATM.

Label Distribution Protocol (LDP) performs label distribution in MPLS environments. LDP provides the following capabilities:

- LDP performs hop-by-hop or dynamic path setup; it does not provide end-to-end switching services.
- LDP assigns labels to routes using the underlying Interior Gateway Protocols (IGP) routing protocols.
- LDP provides constraint-based routing using LDP extensions for traffic engineering.

Finally, LDP is deployed in the core of the network and is one of the key protocols used in MPLS-based Layer 2 and Layer 3 virtual private networks (VPNs).

Feature History for Implementing MPLS LDP

Release	Modification
Release 5.0.0	This feature was introduced.
Release 7.1.1	Multiple MPLS-TE tunnel end points can be enabled on an LER using the TLV 132 function in IS-IS.

Prerequisites for Implementing Cisco MPLS LDP

These prerequisites are required 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.

Information About Implementing Cisco MPLS LDP

To implement MPLS LDP, you should understand these concepts:

Overview of Label Distribution Protocol

LDP performs label distribution in MPLS environments. LDP uses hop-by-hop or dynamic path setup, but does not provide end-to-end switching services. Labels are assigned to routes that are chosen by the underlying IGP routing protocols. The Label Switched Paths (LSPs) that result from the routes, forward labeled traffic across the MPLS backbone to adjacent nodes.

Label Switched Paths

LSPs are created in the network through MPLS. They 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 Control Plane

The control plane enables label switched routers (LSRs) to discover their potential peer routers and to establish LDP sessions with those peers to exchange label binding information.

Related Topics

Configuring LDP Discovery Parameters, on page 14
Configuring LDP Discovery Over a Link, on page 17
Configuring LDP Link: Example, on page 39
Configuring LDP Discovery for Active Targeted Hellos, on page 19

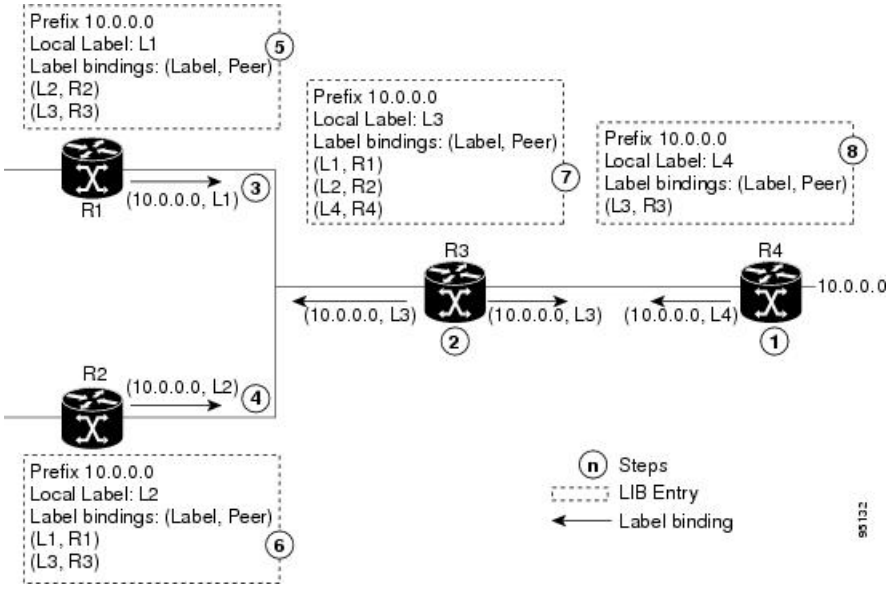
Configuring LDP Discovery for Passive Targeted Hellos, on page 21
Configuring LDP Discovery for Targeted Hellos: Example, on page 40

Exchanging Label Bindings

LDP creates LSPs to perform the hop-by-hop path setup so that MPLS packets can be transferred between the nodes on the MPLS network.

Figure 1: Setting Up Label Switched Paths

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



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.

Related Topics

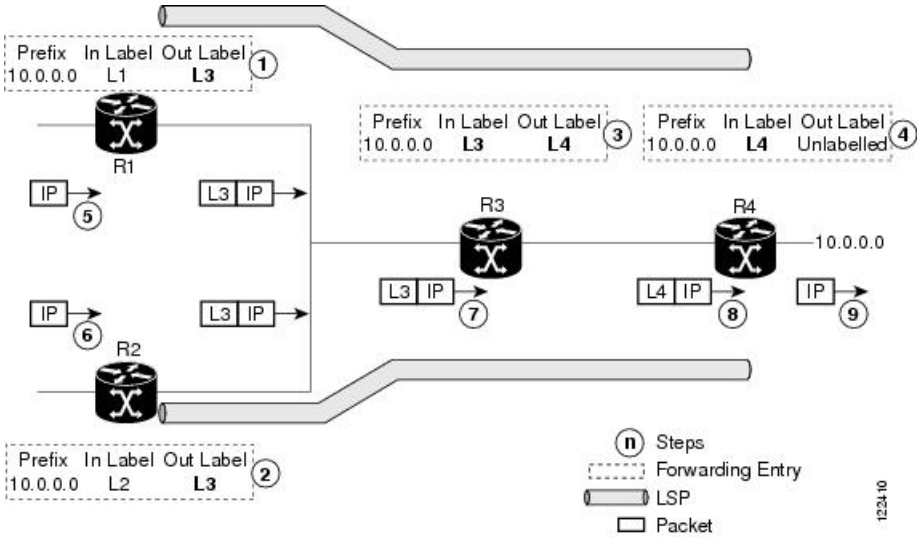
Setting Up LDP Neighbors, on page 24
Configuring LDP Neighbors: Example, on page 41

LDP Forwarding

Once label bindings are learned, the LDP control plane is ready to setup the MPLS forwarding plane as shown in the following figure.

Figure 2: Forwarding Setup

Once label bindings are learned, the LDP control plane is ready to setup the MPLS forwarding plane as shown in this figure.



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.

Note For local labels, only up to 12000 rewrites are supported. If the rewrites exceed this limit, MPLS LSD or MPLS LDP or both the processes may crash.

Related Topics

Setting Up LDP Forwarding, on page 26
Configuring LDP Forwarding: Example, on page 41

LDP 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.

For more information about LDP commands, see *MPLS Label Distribution Protocol Commands* module of the *MPLS Command Reference for Cisco NCS 6000 Series Routers*.

Related Topics

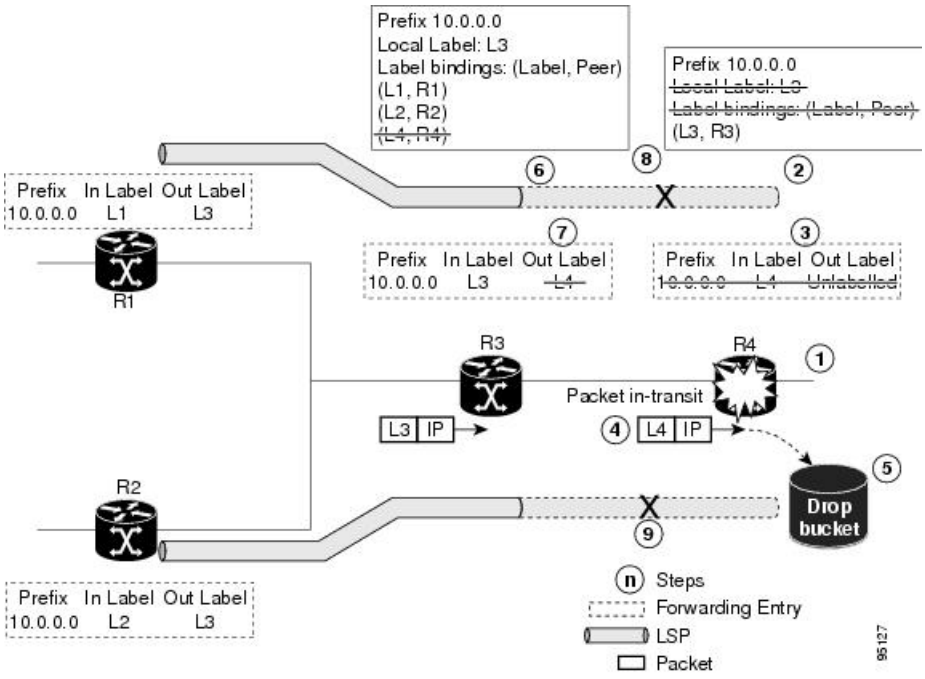
Phases in Graceful Restart, on page 9
Recovery with Graceful-Restart, on page 9
Setting Up LDP NSF Using Graceful Restart, on page 28
Configuring LDP Nonstop Forwarding with Graceful Restart: Example, on page 41

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.

Figure 3: Control Plane Failure

This figure illustrates a control plane failure and shows the process and results of a control plane failure leading to loss of connectivity.



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.

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.

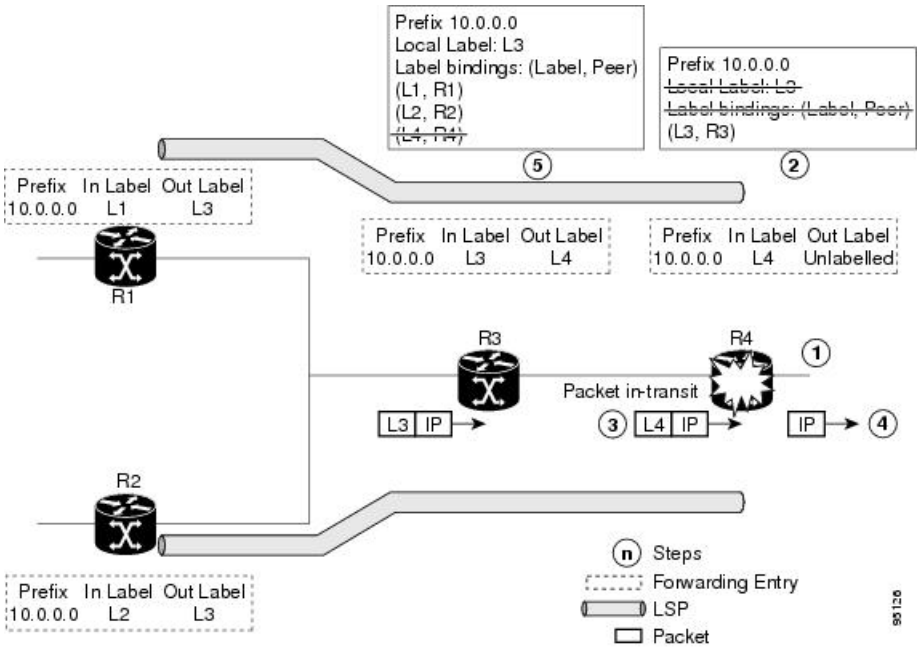
Related Topics

LDP Graceful Restart, on page 7
Recovery with Graceful-Restart, on page 9
Setting Up LDP NSF Using Graceful Restart, on page 28
Configuring LDP Nonstop Forwarding with Graceful Restart: Example, on page 41

Recovery with Graceful-Restart

Figure 4: Recovering with Graceful Restart

This figure illustrates the process of failure recovery using graceful restart.



1. The router R4 LSR control plane restarts.
2. With the control plane restart, LIB is gone but forwarding states installed by R4's LDP control plane are not immediately deleted but are marked as stale.
3. Any in-transit packets from R3 to R4 (still labeled with L4) arrive at R4.
4. The MPLS forwarding plane at R4 performs a successful lookup for the local label L4 as forwarding is still intact. The packet is forwarded accordingly.
5. The router R3 LDP peer detects the failure of the control plane and channel and deletes the label bindings from R4. The peer, however, does not delete the corresponding forwarding states but marks them as stale.
6. At this point there are no forwarding disruptions.
7. The peer also starts the neighbor reconnect timer using the reconnect time value.
8. The established LSPs going toward the router R4 are still intact, and there are no broken LSPs.

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.

Related Topics

LDP Graceful Restart, on page 7
Phases in Graceful Restart, on page 9
Setting Up LDP NSF Using Graceful Restart, on page 28
Configuring LDP Nonstop Forwarding with Graceful Restart: Example, on page 41

Label Advertisement Control (Outbound Filtering)

By default, LDP 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*.

Related Topics

Configuring Label Advertisement Control (Outbound Filtering), on page 23
Configuring Label Advertisement (Outbound Filtering): Example, on page 40

Label Acceptance Control (Inbound Filtering)

By default, 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*.

Note Inbound filtering can also be implemented using an outbound filtering policy; however, you may not be able to implement this system if an LDP peer resides under a different administration domain. When both inbound and outbound filtering options are available, we recommend that you use outbound label filtering.

Related Topics

Configuring Label Acceptance Control (Inbound Filtering), on page 30
Configuring Label Acceptance (Inbound Filtering): Example, on page 42

Local Label Allocation Control

By default, LDP allocates local labels for all prefixes that are not Border Gateway Protocol (BGP) prefixes[1]. This is acceptable when LDP is used for applications other than Layer 3 virtual private networks (L3VPN) core transport. When LDP is used to set up transport LSPs for L3VPN traffic in the core, it is not efficient or even necessary to allocate and advertise local labels for, potentially, thousands of IGP prefixes. In such a case, LDP is typically required to allocate and advertise local label for loopback /32 addresses for PE routers. This

[1] For L3VPN Inter-AS option C, LDP may also be required to assign local labels for some BGP prefixes.

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.

Tip You can configure label allocation using an IP access list to specify a set of prefixes that local labels can allocate and advertise.

Related Topics

Configuring Local Label Allocation Control, on page 31
Configuring Local Label Allocation Control: Example, on page 42

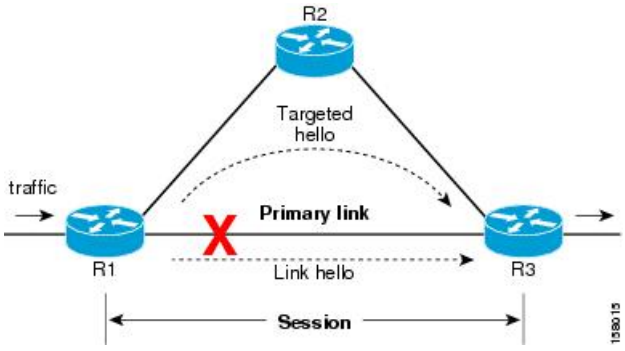
Session Protection

When a link comes up, IP converges earlier and much faster than MPLS LDP and may result in MPLS traffic loss until MPLS convergence. If a link flaps, the LDP session will also flap due to loss of link discovery. LDP session protection minimizes traffic loss, provides faster convergence, and protects existing LDP (link) sessions by means of "parallel" source of targeted discovery hello. An LDP session is kept alive and neighbor label bindings are maintained when links are down. Upon reestablishment of primary link adjacencies, MPLS convergence is expedited as LDP need not relearn the neighbor label bindings.

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 5: Session Protection



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

Related Topics

Configuring Session Protection, on page 32
Configuring LDP Session Protection: Example, on page 43

IGP Synchronization

Lack of synchronization between LDP and 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 synchronizes 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.

In the event of an LDP graceful restart session disconnect, a session is treated as converged as long as the graceful restart neighbor is timed out. Additionally, upon local LDP restart, a checkpointed recovered LDP graceful restart session is used and treated as converged and is given an opportunity to connect and resynchronize.

Under certain circumstances, it might be required to delay declaration of resynchronization 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.

Note The configuration for LDP IGP synchronization resides in respective IGPs (OSPF and IS-IS) and there is no LDP-specific configuration for enabling of this feature. However, there is a specific LDP configuration for IGP sync delay timer.

Related Topics

Configuring LDP IGP Synchronization: OSPF, on page 32
Configuring LDP IGP Synchronization—OSPF: Example, on page 43
Configuring LDP IGP Synchronization: ISIS, on page 33
Configuring LDP IGP Synchronization—ISIS: Example, on page 43

IGP Auto-configuration

To enable LDP on a large number of interfaces, IGP auto-configuration lets you automatically configure LDP on all interfaces associated with a specified IGP interface; for example, when LDP is used for transport in the core network. However, there needs to be one IGP set up to enable LDP auto-configuration.

Typically, LDP assigns and advertises labels for IGP routes and must often be enabled on all active interfaces by an IGP. Without IGP auto-configuration, you must define the set of interfaces under LDP, a procedure that is time-intensive and error-prone.

Note LDP auto-configuration is supported for IPv4 unicast family in the default VRF. The IGP is responsible for verifying and applying the configuration.

You can also disable auto-configuration on a per-interface basis. This permits LDP to enable all IGP interfaces except those that are explicitly disabled and prevents LDP from enabling an interface when LDP auto-configuration is configured under IGP.

Related Topics

Enabling LDP Auto-Configuration for a Specified OSPF Instance, on page 34
Enabling LDP Auto-Configuration in an Area for a Specified OSPF Instance, on page 36
Disabling LDP Auto-Configuration, on page 37
Configuring LDP Auto-Configuration: Example, on page 44

LDP Nonstop Routing

LDP nonstop routing (NSR) functionality makes failures, such as Route Processor (RP) or Distributed Route Processor (DRP) failover, 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
- In-service system upgrade (ISSU)
- 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.

Process failures of active TCP or LDP results in session loss and, as a result, NSR cannot be provided unless RP switchover is configured as a recovery action. For more information about how to configure switchover as a recovery action for NSR, see *Configuring Transports* module in *IP Addresses and Services Configuration Guide for Cisco NCS 6000 Series Routers*.

Related Topics

Configuring LDP Nonstop Routing, on page 38

How to Implement MPLS LDP

A typical MPLS LDP deployment requires coordination among several global neighbor routers. Various configuration tasks are required to implement MPLS LDP :

Configuring LDP Discovery Parameters

Perform this task to configure LDP discovery parameters (which may be crucial for LDP operations).

Note The LDP discovery mechanism is used to discover or locate neighbor nodes.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. [**vrf** *vrf-name*] **router-id** *ip-address lsr-id*
4. **discovery** { **hello** | **targeted-hello** } **holdtime** *seconds*
5. **discovery** { **hello** | **targeted-hello** } **interval** *seconds*
6. **commit**
7. (Optional) **show mpls ldp** [**vrf** *vrf-name*] **parameters**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.
Step 3	[**vrf** *vrf-name*] **router-id** *ip-address lsr-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# router-id 192.168.70.1`	(Optional) Specifies a non-default VRF. Specifies the router ID of the local node. • In Cisco IOS XR software, the router ID is specified as an interface IP address. By default, LDP uses the global router ID (configured by the global router ID process).

Step 4	**discovery** { **hello**	**targeted-hello** } **holdtime** *seconds*
		Example:
		`RP/0/RP0/CPU0:router(config-ldp)# discovery hello holdtime 30`
		`RP/0/RP0/CPU0:router(config-ldp)# discovery targeted-hello holdtime 180`
			Specifies the time that a discovered neighbor is kept without receipt of any subsequent hello messages. The default value for the *seconds* argument is 15 seconds for link hello and 90 seconds for targeted hello messages.

Step 5	**discovery** { **hello**	**targeted-hello** } **interval** *seconds*
		Example:
		`RP/0/RP0/CPU0:router(config-ldp)# discovery hello interval 15`
		`RP/0/RP0/CPU0:router(config-ldp)# discovery targeted-hello interval 20`
			Selects the period of time between the transmission of consecutive hello messages. The default value for the *seconds* argument is 5 seconds for link hello messages and 10 seconds for targeted hello messages.

	Command or Action	Purpose
Step 6	**commit**	

Step 7	(Optional) **show mpls ldp** [**vrf** *vrf-name*] **parameters**
	Example:
	`RP/0/RP0/CPU0:router`
	`# show mpls ldp parameters`
	`RP/0/RP0/CPU0:router`
	`# show mpls ldp vrf red parameters`
		Displays all the current MPLS LDP parameters.
		Displays the LDP parameters for the specified VRF.

Related Topics

LDP Control Plane, on page 4

Configure Label Distribution Protocol Targeted Neighbor

LDP session between LSRs that are not directly connected is known as targeted LDP session. For LDP neighbors which are not directly connected, you must manually configure the LDP neighborship on both the routers.

Configuration Example

This example shows how to configure LDP for non-directly connected routers.

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# mpls ldp
RP/0/RSP0/CPU0:router(config-ldp)# router-id 192.0.2.1
RP/0/RSP0/CPU0:router(config-ldp)# neighbor 198.51.100.1:0 password encrypted 13061E010803
RP/0/RSP0/CPU0:router(config-ldp)# address-family ipv4
RP/0/RSP0/CPU0:router(config-ldp-af)# discovery targeted-hello accept
RP/0/RSP0/CPU0:router(config-ldp-af)# neighbor 198.51.100.1 targeted
RP/0/RSP0/CPU0:router(config-ldp-af)# commit
```

Running Configuration

This section shows the LDP targeted neighbor running configuration.

```
mpls ldp
router-id 192.0.2.1
neighbor 198.51.100.1:0 password encrypted 13061E010803
address-family ipv4
  discovery targeted-hello accept
  neighbor 198.51.100.1 targeted
!
```

Verification

Verify LDP targeted neighbor configuration.

```
RP/0/RSP0/CPU0:router#show mpls ldp discovery
Wed Nov 28 04:30:31.862 UTC

Local LDP Identifier: 192.0.2.1:0
Discovery Sources:
  Targeted Hellos:  <<< targeted hellos based session
    192.0.2.1 -> 198.51.100.1(active/passive), xmit/recv   <<< both transmit and receive
of targeted hellos between the neighbors
      LDP Id: 198.51.100.1:0
          Hold time: 90 sec (local:90 sec, peer:90 sec)
          Established: Nov 28 04:19:55.340 (00:10:36 ago)

RP/0/RSP0/CPU0:router#show mpls ldp neigbhor
Wed Nov 28 04:30:38.272 UTC

Peer LDP Identifier: 198.51.100.1:0
  TCP connection: 198.51.100.1:0:13183 - 192.0.2.1:646; MD5 on
  Graceful Restart: No
  Session Holdtime: 180 sec
  State: Oper; Msgs sent/rcvd: 20/20; Downstream-Unsolicited
  Up time: 00:10:30
  LDP Discovery Sources:
    IPv4: (1)
      Targeted Hello (192.0.2.1 -> 198.51.100.1, active/passive) <<< targeted LDP based
session
    IPv6: (0)
  Addresses bound to this peer:
    IPv4: (4)
     198.51.100.1        10.0.0.1       172.16.0.1       192.168.0.1
    IPv6: (0)
```

Configuring LDP Discovery Over a Link

Perform this task to configure LDP discovery over a link.

Note There is no need to enable LDP globally.

Before you begin

A stable router ID is required at either end of the link to ensure the link discovery (and session setup) is successful. If you do not assign a router ID to the routers, the system will default to the global router ID. Default router IDs are subject to change and may cause an unstable discovery.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. [**vrf** *vrf-name*] **router-id** *ip-address lsr-id*
4. **interface** *type interface-path-id*
5. **commit**
6. (Optional) **show mpls ldp discovery**
7. (Optional) **show mpls ldp vrf** *vrf-name* **discovery**
8. (Optional) **show mpls ldp vrf all discovery summary**

9. (Optional) **show mpls ldp vrf all discovery brief**
10. (Optional) **show mpls ldp vrf all ipv4 discovery summary**
11. (Optional) **show mpls ldp discovery summary all**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.
Step 3	[**vrf** *vrf-name*] **router-id** *ip-address lsr-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# router-id 192.168.70.1`	(Optional) Specifies a non-default VRF. Specifies the router ID of the local node. • In Cisco IOS XR software, the router ID is specified as an interface name or IP address. By default, LDP uses the global router ID (configured by the global router ID process).

Step 4	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-ldp)# interface tunnel-te 12001`
	`RP/0/RP0/CPU0:router(config-ldp-if)#`
		Enters interface configuration mode for the LDP protocol. Interface type must be Tunnel-TE.

Step 5	**commit**	
Step 6	(Optional) **show mpls ldp discovery** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp discovery`	Displays the status of the LDP discovery process. This command, without an interface filter, generates a list of interfaces over which the LDP discovery process is running. The output information contains the state of the link (xmt/rcv hellos), local LDP identifier, the discovered peer's LDP identifier, and holdtime values.
Step 7	(Optional) **show mpls ldp vrf** *vrf-name* **discovery** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf red discovery`	Displays the status of the LDP discovery process for the specified VRF.
Step 8	(Optional) **show mpls ldp vrf all discovery summary** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all discovery summary`	Displays the summarized status of the LDP discovery process for all VRFs.

	Command or Action	Purpose
Step 9	(Optional) **show mpls ldp vrf all discovery brief** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all discovery brief`	Displays the brief status of the LDP discovery process for all VRFs.
Step 10	(Optional) **show mpls ldp vrf all ipv4 discovery summary** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all ipv4 discovery summary`	Displays the summarized status of the LDP discovery process for all VRFs for the IPv4 address family.
Step 11	(Optional) **show mpls ldp discovery summary all** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp discovery summary all`	Displays the aggregate summary across all the LDP discovery processes.

Related Topics

LDP Control Plane, on page 4
Configuring LDP Link: Example, on page 39

Configuring LDP Discovery for Active Targeted Hellos

Perform this task to configure LDP discovery for active targeted hellos.

Note The active side for targeted hellos initiates the unicast hello toward a specific destination.

Before you begin

These prerequisites are required to configure LDP discovery for active targeted hellos:

- Stable router ID is required at either end of the targeted session. If you do not assign a router ID to the routers, the system will default to the global router ID. Please note that default router IDs are subject to change and may cause an unstable discovery.
- One or more MPLS Traffic Engineering tunnels are established between non-directly connected LSRs.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. [**vrf** *vrf-name*] **router-id** *ip-address* *lsr-id*
4. **interface** *type interface-path-id*
5. **commit**

6. (Optional) **show mpls ldp discovery**
7. (Optional) **show mpls ldp vrf** *vrf-name* **discovery**
8. (Optional) **show mpls ldp vrf all discovery summary**
9. (Optional) **show mpls ldp vrf all discovery brief**
10. (Optional) **show mpls ldp vrf all ipv4 discovery summary**
11. (Optional) **show mpls ldp discovery summary all**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.
Step 3	[**vrf** *vrf-name*] **router-id** *ip-address lsr-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# router-id 192.168.70.1`	(Optional) Specifies a non-default VRF. Specifies the router ID of the local node. In Cisco IOS XR software, the router ID is specified as an interface name or IP address or LSR ID. By default, LDP uses the global router ID (configured by global router ID process).
Step 4	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# interface tunnel-te 12001`	Enters interface configuration mode for the LDP protocol.
Step 5	**commit**	
Step 6	(Optional) **show mpls ldp discovery** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp discovery`	Displays the status of the LDP discovery process. This command, without an interface filter, generates a list of interfaces over which the LDP discovery process is running. The output information contains the state of the link (xmt/rcv hellos), local LDP identifier, the discovered peer's LDP identifier, and holdtime values.
Step 7	(Optional) **show mpls ldp vrf** *vrf-name* **discovery** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf red discovery`	Displays the status of the LDP discovery process for the specified VRF.
Step 8	(Optional) **show mpls ldp vrf all discovery summary** **Example:**	Displays the summarized status of the LDP discovery process for all VRFs.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# show mpls ldp vrf all discovery summary`	
Step 9	(Optional) **show mpls ldp vrf all discovery brief** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all discovery brief`	Displays the brief status of the LDP discovery process for all VRFs.
Step 10	(Optional) **show mpls ldp vrf all ipv4 discovery summary** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all ipv4 discovery summary`	Displays the summarized status of the LDP discovery process for all VRFs for the IPv4 address family.
Step 11	(Optional) **show mpls ldp discovery summary all** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp discovery summary all`	Displays the aggregate summary across all the LDP discovery processes.

Related Topics

LDP Control Plane, on page 4
Configuring LDP Discovery for Targeted Hellos: Example, on page 40

Configuring LDP Discovery for Passive Targeted Hellos

Perform this task to configure LDP discovery for passive targeted hellos.

A passive side for targeted hello is the destination router (tunnel tail), which passively waits for an incoming hello message. Because targeted hellos are unicast, the passive side waits for an incoming hello message to respond with hello toward its discovered neighbor.

Before you begin

Stable router ID is required at either end of the link to ensure that the link discovery (and session setup) 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 and may cause an unstable discovery.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. [**vrf** *vrf-name*] **router-id** *ip-address* *lsr-id*
4. **discovery targeted-hello accept**
5. **commit**
6. (Optional) **show mpls ldp discovery**

7. (Optional) **show mpls ldp vrf** *vrf-name* **discovery**
8. (Optional) **show mpls ldp vrf all discovery summary**
9. (Optional) **show mpls ldp vrf all discovery brief**
10. (Optional) **show mpls ldp vrf all ipv4 discovery summary**
11. (Optional) **show mpls ldp discovery summary all**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.
Step 3	[**vrf** *vrf-name*] **router-id** *ip-address lsr-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# router-id 192.168.70.1`	(Optional) Specifies a non-default VRF. Specifies the router ID of the local node. • In Cisco IOS XR software, the router ID is specified as an interface IP address or LSR ID. By default, LDP uses the global router ID (configured by global router ID process).
Step 4	**discovery targeted-hello accept** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# discovery targeted-hello accept`	Directs the system to accept targeted hello messages from any source and activates passive mode on the LSR for targeted hello acceptance. • This command is executed on the receiver node (with respect to a given MPLS TE tunnel). • You can control the targeted-hello acceptance using the **discovery targeted-hello accept** command.
Step 5	**commit**	
Step 6	(Optional) **show mpls ldp discovery** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp discovery`	Displays the status of the LDP discovery process. This command, without an interface filter, generates a list of interfaces over which the LDP discovery process is running. The output information contains the state of the link (xmt/rcv hellos), local LDP identifier, the discovered peer's LDP identifier, and holdtime values.
Step 7	(Optional) **show mpls ldp vrf** *vrf-name* **discovery** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf red discovery`	Displays the status of the LDP discovery process for the specified VRF.
Step 8	(Optional) **show mpls ldp vrf all discovery summary** **Example:**	Displays the summarized status of the LDP discovery process for all VRFs.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# show mpls ldp vrf all discovery summary`	
Step 9	(Optional) **show mpls ldp vrf all discovery brief** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all discovery brief`	Displays the brief status of the LDP discovery process for all VRFs.
Step 10	(Optional) **show mpls ldp vrf all ipv4 discovery summary** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp vrf all ipv4 discovery summary`	Displays the summarized status of the LDP discovery process for all VRFs for the IPv4 address family.
Step 11	(Optional) **show mpls ldp discovery summary all** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp discovery summary all`	Displays the aggregate summary across all the LDP discovery processes.

Related Topics

LDP Control Plane, on page 4
Configuring LDP Discovery for Targeted Hellos: Example, on page 40

Configuring Label Advertisement Control (Outbound Filtering)

Perform this task to configure label advertisement (outbound filtering).

By default, a label switched router (LSR) advertises all incoming label prefixes to each neighboring router. 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.

Note Prefixes and peers advertised selectively are defined in the access list.

Before you begin

Before configuring label advertisement, enable LDP and configure an access list.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **label advertise** { **disable** | **for** *prefix-acl* [**to** *peer-acl*] | **interface** *type interface-path-id* }

4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.

Step 3	**label advertise** { **disable** | **for** *prefix-acl* [**to** *peer-acl*] | **interface** *type interface-path-id* }
	Example:
	`RP/0/RP0/CPU0:router(config-ldp)# label advertise interface POS 0/1/0/0`
	`RP/0/RP0/CPU0:router(config-ldp)# for pfx_acl1 to peer_acl1`
		Configures label advertisement by specifying one of the following options:
		disable
		Disables label advertisement to all peers for all prefixes (if there are no other conflicting rules).
		interface
		Specifies an interface for label advertisement of an interface address.
		for ***prefix-acl***
		to ***peer-acl***
		Specifies neighbors to advertise and receive label advertisements.

Step 4	**commit**	

Related Topics

Label Advertisement Control (Outbound Filtering), on page 11
Configuring Label Advertisement (Outbound Filtering): Example, on page 40

Setting Up LDP Neighbors

Perform this task to set up LDP neighbors.

Before you begin

Stable router ID is required at either end of the link to ensure the link discovery (and session setup) is successful. If you do not assign a router ID to the routers, the system will default to the global router ID. Default router IDs are subject to change and may cause an unstable discovery.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **interface** *type interface-path-id*
4. **discovery transport-address** [*ip-address* \| **interface**]

5. **exit**
6. **holdtime** *seconds*
7. **neighbor** *ip-address* **password** [*encryption*] *password*
8. **backoff** *initial maximum*
9. **commit**
10. (Optional) **show mpls ldp neighbor**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.
Step 3	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# interface POS 0/1/0/0`	Enters interface configuration mode for the LDP protocol.
Step 4	**discovery transport-address** [*ip-address* | **interface**] **Example:** or `RP/0/RP0/CPU0:router(config-ldp-if-af)# discovery transport-address interface`	Provides an alternative transport address for a TCP connection. • Default transport address advertised by an LSR (for TCP connections) to its peer is the router ID. • Transport address configuration is applied for a given LDP-enabled interface. • If the interface version of the command is used, the configured IP address of the interface is passed to its neighbors as the transport address.
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:router(config-ldp-if)# exit`	Exits the current configuration mode.
Step 6	**holdtime** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# holdtime 30`	Changes the time for which an LDP session is maintained in the absence of LDP messages from the peer. • Outgoing keepalive interval is adjusted accordingly (to make three keepalives in a given holdtime) with a change in session holdtime value. • Session holdtime is also exchanged when the session is established.

	Command or Action	Purpose
		• In this example holdtime is set to 30 seconds, which causes the peer session to timeout in 30 seconds, as well as transmitting outgoing keepalive messages toward the peer every 10 seconds.
Step 7	**neighbor** *ip-address* **password** [*encryption*] *password* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# neighbor 192.168.2.44 password secretpasswd`	Configures password authentication (using the TCP MD5 option) for a given neighbor.
Step 8	**backoff** *initial maximum* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# backoff 10 20`	Configures the parameters for the LDP backoff mechanism. The LDP backoff mechanism prevents two incompatibly configured LSRs from engaging in an unthrottled sequence of session setup failures. If a session setup attempt fails due to such incompatibility, each LSR delays its next attempt (backs off), increasing the delay exponentially with each successive failure until the maximum backoff delay is reached.
Step 9	**commit**	
Step 10	(Optional) **show mpls ldp neighbor** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp neighbor`	Displays the status of the LDP session with its neighbors. This command can be run with various filters as well as with the brief option.

Related Topics

Configuring LDP Neighbors: Example, on page 41

Setting Up LDP Forwarding

Perform this task to set up LDP forwarding.

By default, the LDP control plane implements the penultimate hop popping (PHOP) mechanism. The PHOP mechanism requires that label switched routers use the implicit-null label as a local label for the given Forwarding Equivalence Class (FEC) for which LSR is the penultimate hop. Although PHOP has certain advantages, it may be required to extend LSP up to the ultimate hop under certain circumstances (for example, to propagate MPL QoS). This is done using a special local label (explicit-null) advertised to the peers after which the peers use this label when forwarding traffic toward the ultimate hop (egress LSR).

Before you begin

Stable router ID is required at either end of the link to ensure the link discovery (and session setup) is successful. If you do not assign a router ID to the routers, the system will default to the global router ID. Default router IDs are subject to change and may cause an unstable discovery.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **explicit-null**
4. **commit**
5. (Optional) **show mpls ldp forwarding**
6. (Optional) **show mpls forwarding**
7. (Optional) **ping** *ip-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.
Step 3	**explicit-null** **Example:** `RP/0/RP0/CPU0:router(config-ldp-af)# explicit-null`	Causes a router to advertise an explicit null label in situations where it normally advertises an implicit null label (for example, to enable an ultimate-hop disposition instead of PHOP).
Step 4	**commit**	
Step 5	(Optional) **show mpls ldp forwarding** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp forwarding`	Displays the MPLS LDP view of installed forwarding states (rewrites). **Note** For local labels, only up to 12000 rewrites are supported. If the rewrites exceed this limit, MPLS LSD or MPLS LDP or both the processes may crash.
Step 6	(Optional) **show mpls forwarding** **Example:** `RP/0/RP0/CPU0:router# show mpls forwarding`	Displays a global view of all MPLS installed forwarding states (rewrites) by various applications (LDP, TE, and static).
Step 7	(Optional) **ping** *ip-address* **Example:** `RP/0/RP0/CPU0:router# ping 192.168.2.55`	Checks for connectivity to a particular IP address (going through MPLS LSP as shown in the **show mpls forwarding** command).

Related Topics

LDP Forwarding, on page 6
Configuring LDP Forwarding: Example, on page 41

Setting Up LDP NSF Using Graceful Restart

Perform this task to set up NSF using LDP graceful restart.

LDP graceful restart is a way to enable NSF for LDP. The correct way to set up NSF using LDP graceful restart is to bring up LDP neighbors (link or targeted) with additional configuration related to graceful restart.

Before you begin

Stable router ID is required at either end of the link to ensure the link discovery (and session setup) is successful. If you do not assign a router ID to the routers, the system will default to the global router ID. Default router IDs are subject to change and may cause an unstable discovery.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **interface** *type interface-path-id*
4. **exit**
5. **graceful-restart**
6. **graceful-restart forwarding-state-holdtime** *seconds*
7. **graceful-restart reconnect-timeout** *seconds*
8. **commit**
9. (Optional) **show mpls ldp parameters**
10. (Optional) **show mpls ldp neighbor**
11. (Optional) **show mpls ldp graceful-restart**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters MPLS LDP configuration mode.

Step 3	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-ldp)# interface POS 0/1/0/0`
	`RP/0/RP0/CPU0:router(config-ldp-if)#`
		Enters interface configuration mode for the LDP protocol.

Step 4	**exit** **Example:** `RP/0/RP0/CPU0:router(config-ldp-if)# exit`	Exits the current configuration mode.

	Command or Action	Purpose
Step 5	**graceful-restart** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# graceful-restart`	Enables the LDP graceful restart feature.
Step 6	**graceful-restart forwarding-state-holdtime** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# graceful-restart forwarding-state-holdtime 180`	Specifies the length of time that forwarding can keep LDP-installed forwarding states and rewrites, and specifies wh en the LDP control plane restarts. • After restart of the control plane, when the forwarding state holdtime expires, any previously installed LDP forwarding state or rewrite that is not yet refreshed is deleted from the forwarding. • Recovery time sent after restart is computed as the current remaining value of the forwarding state hold timer.
Step 7	**graceful-restart reconnect-timeout** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# graceful-restart reconnect-timeout 169`	Specifies the length of time a neighbor waits before restarting the node to reconnect before declaring an earlier graceful restart session as down. This command is used to start a timer on the peer (upon a neighbor restart). This timer is referred to as *Neighbor Liveness* timer.
Step 8	**commit**	

Step 9	(Optional) **show mpls ldp parameters**
	Example:
	`RP/0/RP0/CPU0:router`
	`# show mpls ldp parameters`
		Displays all the current MPLS LDP parameters.

Step 10	(Optional) **show mpls ldp neighbor** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp neighbor`	Displays the status of the LDP session with its neighbors. This command can be run with various filters as well as with the brief option.
Step 11	(Optional) **show mpls ldp graceful-restart** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp graceful-restart`	Displays the status of the LDP graceful restart feature. The output of this command not only shows states of different graceful restart timers, but also a list of graceful restart neighbors, their state, and reconnect count.

Related Topics

LDP Graceful Restart, on page 7
Phases in Graceful Restart, on page 9

Recovery with Graceful-Restart, on page 9
Configuring LDP Nonstop Forwarding with Graceful Restart: Example, on page 41

Configuring Label Acceptance Control (Inbound Filtering)

Perform this task to configure LDP inbound label filtering.

Note By default, there is no inbound label filtering performed by LDP and thus an LSR accepts (and retains) all remote label bindings from all peers.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **label accept for** *prefix-acl* **from** *ip-address*
4. [**vrf** *vrf-name*] **address-family** { **ipv4**}
5. **label remote accept from** *ldp-id* **for** *prefix-acl*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters the MPLS LDP configuration mode.

Step 3	**label accept for** *prefix-acl* **from** *ip-address*
	Example:
	`RP/0/RP0/CPU0:router(config-ldp)# label accept for pfx_acl_1 from 192.168.1.1`
	`RP/0/RP0/CPU0:router(config-ldp)# label accept for pfx_acl_2 from 192.168.2.2`
		Configures inbound label acceptance for prefixes specified by prefix-acl from neighbor (as specified by its IP address).

Step 4	[**vrf** *vrf-name*] **address-family** { **ipv4**}
	Example:
	`RP/0/RP0/CPU0:router(config-ldp)# address-family ipv4`
	`RP/0/RP0/CPU0:router(config-ldp)# address-family ipv6`
		(Optional) Specifies a non-default VRF.
		Enables the LDP IPv4 or IPv6 address family.

	Command or Action	Purpose
Step 5	**label remote accept from** *ldp-id* **for** *prefix-acl* **Example:** `RP/0/RP0/CPU0:router(config-ldp-af)# label remote accept from 192.168.1.1:0 for pfx_acl_1`	Configures inbound label acceptance control for prefixes specified by prefix-acl from neighbor (as specified by its LDP ID).
Step 6	**commit**	

Related Topics

Label Acceptance Control (Inbound Filtering), on page 11
Configuring Label Acceptance (Inbound Filtering): Example, on page 42

Configuring Local Label Allocation Control

Perform this task to configure label allocation control.

Note By default, local label allocation control is disabled and all non-BGP prefixes are assigned local labels.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **label allocate for** *prefix-acl*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters the MPLS LDP configuration mode.
Step 3	**label allocate for** *prefix-acl* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# label allocate for pfx_acl_1`	Configures label allocation control for prefixes as specified by prefix-acl.
Step 4	**commit**	

Related Topics

Local Label Allocation Control, on page 11
Configuring Local Label Allocation Control: Example, on page 42

Configuring Session Protection

Perform this task to configure LDP session protection.

By default, there is no protection is done for link sessions by means of targeted hellos.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **session protection** [**for** *peer-acl*] [**duration** *seconds*]
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters the MPLS LDP configuration mode.
Step 3	**session protection** [**for** *peer-acl*] [**duration** *seconds*] **Example:** `RP/0/RP0/CPU0:router(config-ldp)# session protection for peer_acl_1 duration 60`	Configures LDP session protection for peers specified by peer-acl with a maximum duration, in seconds.
Step 4	**commit**	

Related Topics

Session Protection, on page 12
Configuring LDP Session Protection: Example, on page 43

Configuring LDP IGP Synchronization: OSPF

Perform this task to configure LDP IGP Synchronization under OSPF.

Note By default, there is no synchronization between LDP and IGPs.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. Use one of the following commands:
   - **mpls ldp sync**
   - **area** *area-id* **mpls ldp sync**
   - **area** *area-id* **interface** *name* **mpls ldp sync**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router ospf** *process-name* **Example:** `RP/0/RP0/CPU0:router(config)# router ospf 100`	Identifies the OSPF routing process and enters OSPF configuration mode.
Step 3	Use one of the following commands: • **mpls ldp sync** • **area** *area-id* **mpls ldp sync** • **area** *area-id* **interface** *name* **mpls ldp sync** **Example:** `RP/0/RP0/CPU0:router(config-ospf)# mpls ldp sync`	Enables LDP IGP synchronization on an interface.
Step 4	**commit**	

Related Topics

IGP Synchronization, on page 13
Configuring LDP IGP Synchronization—OSPF: Example, on page 43

Configuring LDP IGP Synchronization: ISIS

Perform this task to configure LDP IGP Synchronization under ISIS.

Note By default, there is no synchronization between LDP and ISIS.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **interface** *type interface-path-id*

4. **address-family** {**ipv4** } **unicast**
5. **mpls ldp sync**
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**router isis** *instance-id*
	Example:
	`RP/0/RP0/CPU0:router(config)# router isis 100`
	`RP/0/RP0/CPU0:router(config-isis)#`
		Enables the Intermediate System-to-Intermediate System (IS-IS) routing protocol and defines an IS-IS instance.

Step 3	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-isis)# interface POS 0/2/0/0`
	`RP/0/RP0/CPU0:router(config-isis-if)#`
		Configures the IS-IS protocol on an interface and enters ISIS interface configuration mode.

Step 4	**address-family** {**ipv4** } **unicast**
	Example:
	`RP/0/RP0/CPU0:router(config-isis-if)#`
	`address-family ipv4 unicast`
	`RP/0/RP0/CPU0:router(config-isis-if-af)#`
		Enters address family configuration mode for configuring IS-IS routing for a standard IP version 4 (IPv4) address prefix.

Step 5	**mpls ldp sync** **Example:** `RP/0/RP0/CPU0:router(config-isis-if-af)# mpls ldp sync`	Enables LDP IGP synchronization.
Step 6	**commit**	

Related Topics

IGP Synchronization, on page 13
Configuring LDP IGP Synchronization—ISIS: Example, on page 43

Enabling LDP Auto-Configuration for a Specified OSPF Instance

Perform this task to enable IGP auto-configuration globally for a specified OSPF process name.

You can disable auto-configuration on a per-interface basis. This lets LDP enable all IGP interfaces except those that are explicitly disabled.

Note This feature is supported for IPv4 unicast family in default VRF only.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **mpls ldp auto-config**
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**router ospf** *process-name*
	Example:
	`RP/0/RP0/CPU0:router(config)# router ospf 190`
	`RP/0/RP0/CPU0:router(config-ospf)#`
		Enters a uniquely identifiable OSPF routing process. The process name is any alphanumeric string no longer than 40 characters without spaces.

Step 3	**mpls ldp auto-config** **Example:** `RP/0/RP0/CPU0:router(config-ospf)# mpls ldp auto-config`	Enables LDP auto-configuration.
Step 4	**area** *area-id* **Example:** `RP/0/RP0/CPU0:router(config-ospf)# area 8`	Configures an OSPF area and identifier. ***area-id*** Either a decimal value or an IP address.
Step 5	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar)# interface pos 0/6/0/0`	Enables LDP auto-configuration on the specified interface. **Note** LDP configurable limit for maximum number of interfaces does not apply to IGP auto-configuration interfaces.
Step 6	**commit**	

Related Topics

IGP Auto-configuration, on page 13
Configuring LDP Auto-Configuration: Example, on page 44
Disabling LDP Auto-Configuration, on page 37

Enabling LDP Auto-Configuration in an Area for a Specified OSPF Instance

Perform this task to enable IGP auto-configuration in a defined area with a specified OSPF process name.

You can disable auto-configuration on a per-interface basis. This lets LDP enable all IGP interfaces except those that are explicitly disabled.

Note This feature is supported for IPv4 unicast family in default VRF only.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **mpls ldp auto-config**
5. **interface** *type interface-path-id*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**router ospf** *process-name*
	Example:
	`RP/0/RP0/CPU0:router(config)# router ospf 100`
	`RP/0/RP0/CPU0:router(config-ospf)#`
		Enters a uniquely identifiable OSPF routing process. The process name is any alphanumeric string no longer than 40 characters without spaces.

Step 3	**area** *area-id*
	Example:
	`RP/0/RP0/CPU0:router(config-ospf)# area 8`
	`RP/0/RP0/CPU0:router(config-ospf-ar)#`
		Configures an OSPF area and identifier.
		area-id
		Either a decimal value or an IP address.

Step 4	**mpls ldp auto-config** **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar)# mpls ldp auto-config`	Enables LDP auto-configuration.

Step 5	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-ospf-ar)# interface pos 0/6/0/0`
	`RP/0/RP0/CPU0:router(config-ospf-ar-if)`
		Enables LDP auto-configuration on the specified interface. The LDP configurable limit for maximum number of interfaces does not apply to IGP auto-config interfaces.

	Command or Action	Purpose
Step 6	**commit**	

Related Topics

IGP Auto-configuration, on page 13
Configuring LDP Auto-Configuration: Example, on page 44
Disabling LDP Auto-Configuration, on page 37

Disabling LDP Auto-Configuration

Perform this task to disable IGP auto-configuration.

You can disable auto-configuration on a per-interface basis. This lets LDP enable all IGP interfaces except those that are explicitly disabled.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **interface** *type interface-path-id*
4. **igp auto-config disable**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**mpls ldp**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls ldp`
	`RP/0/RP0/CPU0:router(config-ldp)#`
		Enters the MPLS LDP configuration mode.

Step 3	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-ldp)# interface pos 0/6/0/0`	Enters interface configuration mode and configures an interface.
Step 4	**igp auto-config disable** **Example:** `RP/0/RP0/CPU0:router(config-ldp-if)# igp auto-config disable`	Disables auto-configuration on the specified interface.
Step 5	**commit**	

Related Topics

IGP Auto-configuration, on page 13
Configuring LDP Auto-Configuration: Example, on page 44

Configuring LDP Nonstop Routing

Perform this task to configure LDP NSR.

Note By default, NSR is globally-enabled on all LDP sessions except AToM.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **nsr**
4. **commit**
5. (Optional) **show mpls ldp nsr statistics**
6. (Optional) **show mpls ldp nsr summary**
7. (Optional) **show mpls ldp nsr pending**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls ldp** **Example:** `RP/0/RP0/CPU0:router(config)# mpls ldp`	Enters the MPLS LDP configuration mode.
Step 3	**nsr** **Example:** `RP/0/RP0/CPU0:router(config-ldp)# nsr`	Enables LDP nonstop routing.
Step 4	**commit**	
Step 5	(Optional) **show mpls ldp nsr statistics** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp nsr statistics`	Displays MPLS LDP NSR statistics.
Step 6	(Optional) **show mpls ldp nsr summary** **Example:**	Displays MPLS LDP NSR summarized information.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# show mpls ldp nsr summary`	
Step 7	(Optional) **show mpls ldp nsr pending** **Example:** `RP/0/RP0/CPU0:router# show mpls ldp nsr pending`	Displays MPLS LDP NSR pending information.

Related Topics

LDP Nonstop Routing, on page 14

Configuration Examples for Implementing MPLS LDP

These configuration examples are provided to implement LDP:

Configuring LDP with Graceful Restart: Example

The example shows how to enable LDP with graceful restart on the POS interface 0/2/0/0.

```
mpls ldp
 graceful-restart
 interface pos0/2/0/0
!
```

Configuring LDP Discovery: Example

The example shows how to configure LDP discovery parameters.

```
mpls ldp
 router-id 192.168.70.1
 discovery hello holdtime 15
 discovery hello interval 5
!

show mpls ldp parameters
show mpls ldp discovery
```

Configuring LDP Link: Example

The example shows how to configure LDP link parameters.

```
mpls ldp
 interface pos 0/1/0/0
 !
!
```

```
show mpls ldp discovery
```

Related Topics

Configuring LDP Discovery Over a Link, on page 17
LDP Control Plane, on page 4

Configuring LDP Discovery for Targeted Hellos: Example

The examples show how to configure LDP Discovery to accept targeted hello messages.

Active (tunnel head)

```
mpls ldp
 router-id 192.168.70.1
 interface tunnel-te 12001
 !
!
```

Passive (tunnel tail)

```
mpls ldp
 router-id 192.168.70.2
 discovery targeted-hello accept
!
```

Related Topics

Configuring LDP Discovery for Active Targeted Hellos, on page 19
Configuring LDP Discovery for Passive Targeted Hellos, on page 21
LDP Control Plane, on page 4

Configuring Label Advertisement (Outbound Filtering): Example

The example shows how to configure LDP label advertisement control.

```
 mpls ldp
    label
        advertise
            disable
            for pfx_acl_1 to peer_acl_1
            for pfx_acl_2 to peer_acl_2
            for pfx_acl_3
            interface POS 0/1/0/0
            interface POS 0/2/0/0
        !
     !
!
ipv4 access-list pfx_acl_1
    10 permit ip host 1.0.0.0 any
!
ipv4 access-list pfx_acl_2
    10 permit ip host 2.0.0.0 any
```

```
!
ipv4 access-list peer_acl_1
    10 permit ip host 1.1.1.1 any
    20 permit ip host 1.1.1.2 any
!
ipv4 access-list peer_acl_2
    10 permit ip host 2.2.2.2 any
!

show mpls ldp binding
```

Related Topics

Configuring Label Advertisement Control (Outbound Filtering), on page 23
Label Advertisement Control (Outbound Filtering), on page 11

Configuring LDP Neighbors: Example

The example shows how to disable label advertisement.

```
mpls ldp
    router-id 192.168.70.1
    neighbor 1.1.1.1 password encrypted 110A1016141E
    neighbor 2.2.2.2 implicit-withdraw
!
```

Related Topics

Setting Up LDP Neighbors, on page 24

Configuring LDP Forwarding: Example

The example shows how to configure LDP forwarding.

```
mpls ldp
 address-family ipv4
 label local advertise explicit-null
!

show mpls ldp forwarding
show mpls forwarding
```

Related Topics

Setting Up LDP Forwarding, on page 26
LDP Forwarding, on page 6

Configuring LDP Nonstop Forwarding with Graceful Restart: Example

The example shows how to configure LDP nonstop forwarding with graceful restart.

```
mpls ldp
log
graceful-restart
!
```

```
 graceful-restart
 graceful-restart forwarding state-holdtime 180
 graceful-restart reconnect-timeout 15
 interface pos0/1/0/0
!

show mpls ldp graceful-restart
show mpls ldp neighbor gr
show mpls ldp forwarding
show mpls forwarding
```

Related Topics

Setting Up LDP NSF Using Graceful Restart, on page 28
LDP Graceful Restart, on page 7
Phases in Graceful Restart, on page 9
Recovery with Graceful-Restart, on page 9

Configuring Label Acceptance (Inbound Filtering): Example

The example shows how to configure inbound label filtering.

```
mpls ldp
 label
accept
 for pfx_acl_2 from 192.168.2.2
!
 !
!
```

```
mpls ldp
 address-family ipv4
  label remote accept from 192.168.1.1:0 for pfx_acl_2
  !
 !
!
```

Related Topics

Configuring Label Acceptance Control (Inbound Filtering), on page 30
Label Acceptance Control (Inbound Filtering), on page 11

Configuring Local Label Allocation Control: Example

The example shows how to configure local label allocation control.

```
mpls ldp
 label
allocate for pfx_acl_1
 !
!
```

Related Topics

Configuring Local Label Allocation Control, on page 31
Local Label Allocation Control, on page 11

Configuring LDP Session Protection: Example

The example shows how to configure session protection.

```
mpls ldp
 session protection duration

!
```

Related Topics

Configuring Session Protection, on page 32
Session Protection, on page 12

Configuring LDP IGP Synchronization—OSPF: Example

The example shows how to configure LDP IGP synchronization for OSPF.

```
router ospf 100
mpls ldp sync
!
mpls ldp
 igp sync delay 30
!
```

Related Topics

Configuring LDP IGP Synchronization: OSPF, on page 32
IGP Synchronization, on page 13

Configuring LDP IGP Synchronization—ISIS: Example

The example shows how to configure LDP IGP synchronization.

```
router isis 100
 interface POS 0/2/0/0
address-family ipv4 unicast
mpls ldp sync
!
 !
!
mpls ldp
 igp sync delay 30
!
```

Related Topics

Configuring LDP IGP Synchronization: ISIS, on page 33
IGP Synchronization, on page 13

Configuring LDP Auto-Configuration: Example

The example shows how to configure the IGP auto-configuration feature globally for a specific OSPF interface ID.

```
router ospf 100
 mpls ldp auto-config
 area 0
  interface pos 1/1/1/1
```

The example shows how to configure the IGP auto-configuration feature on a given area for a given OSPF interface ID.

```
router ospf 100
 area 0
  mpls ldp auto-config
  interface pos 1/1/1/1
```

Related Topics

Enabling LDP Auto-Configuration for a Specified OSPF Instance, on page 34
Enabling LDP Auto-Configuration in an Area for a Specified OSPF Instance, on page 36
Disabling LDP Auto-Configuration, on page 37
IGP Auto-configuration, on page 13

Additional References

For additional information related to Implementing MPLS Label Distribution Protocol, refer to the following references:

Related Documents

Related Topic	Document Title
LDP Commands	*MPLS Label Distribution Protocol Commands* module in *MPLS Command Reference for Cisco NCS 6000 Series Routers*.

Standards

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

MIBs

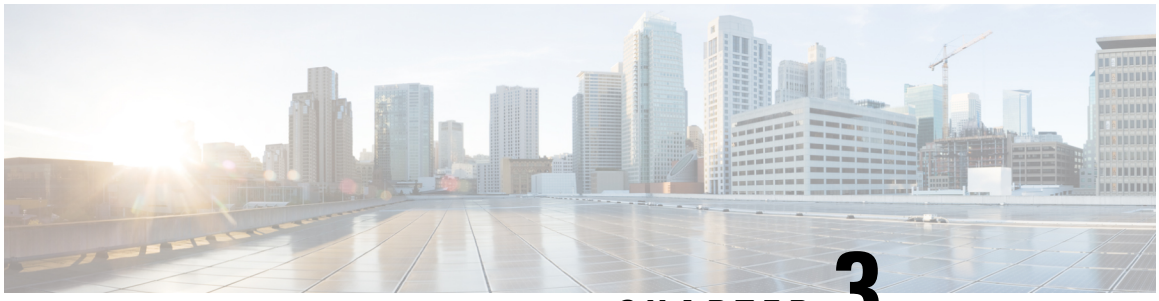
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs **Note** **Not all supported RFCs are listed.**	Title
RFC 3031	*Multiprotocol Label Switching Architecture*
RFC 3036	*LDP Specification*
RFC 3037	*LDP Applicability*
RFC 3478	*Graceful Restart Mechanism for Label Distribution Protocol*
RFC 3815	*Definitions of Managed Objects for MPLS LDP*
RFC 5036	*Label Distribution and Management* *Downstream on Demand Label Advertisement*
RFC 5286	*Basic Specification for IP Fast Reroute: Loop-Free Alternates*

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



CHAPTER 3

Implementing RSVP for MPLS-TE

The Multiprotocol Label Switching (MPLS) is a standards-based solution, driven by the Internet Engineering Task Force (IETF), devised to convert the Internet and IP backbones from best-effort networks into business-class transport media.

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.

RSVP provides a secure method to control quality-of-service (QoS) access to a network.

MPLS Traffic Engineering (MPLS-TE) RSVP to signal label switched paths (LSPs).

Feature History for Implementing RSVP for MPLS-TE

Release	Modification
Release 5.0.0	This feature was introduced.

Prerequisites for Implementing RSVP for MPLS-TE

These prerequisites are required to implement RSVP for MPLS-TE :

- 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.
- Either a composite mini-image plus an MPLS package, or a full image, must be installed.

Information About Implementing RSVP for MPLS-TE

To implement MPLS RSVP, you must understand the these concepts:

Related Topics

How to Implement RSVP Authentication, on page 65

Overview of RSVP for MPLS-TE

RSVP is a network control protocol that enables Internet applications to signal LSPs for MPLS-TE . The RSVP implementation is compliant with the IETF RFC 2205, RFC 3209.

RSVP is automatically enabled on interfaces on which MPLS-TE is configured. For MPLS-TE LSPs with nonzero 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 Refresh Reduction, defined in RFC 2961, includes support for reliable messages and summary refresh messages. Reliable messages are retransmitted rapidly if the message is lost. Because each summary refresh message contains information to refresh multiple states, this greatly reduces the amount of messaging needed to refresh states. For refresh reduction to be used between two routers, it must be enabled on both routers. Refresh Reduction is enabled by default.

Message rate limiting for RSVP allows you to set a maximum threshold on the rate at which RSVP messages are sent on an interface. Message rate limiting is disabled by default.

The process that implements RSVP is restartable. A software upgrade, process placement or process failure of RSVP or any of its collaborators, has been designed to ensure Nonstop Forwarding (NSF) of the data plane.

RSVP supports graceful restart, which is compliant with RFC 3473. It follows the procedures that apply when the node reestablishes communication with the neighbor's control plane within a configured restart time.

It is important to note that RSVP is not a routing protocol. RSVP works in conjunction with routing protocols and installs the equivalent of dynamic access lists along the routes that routing protocols calculate. Because of this, implementing RSVP in an existing network does not require migration to a new routing protocol.

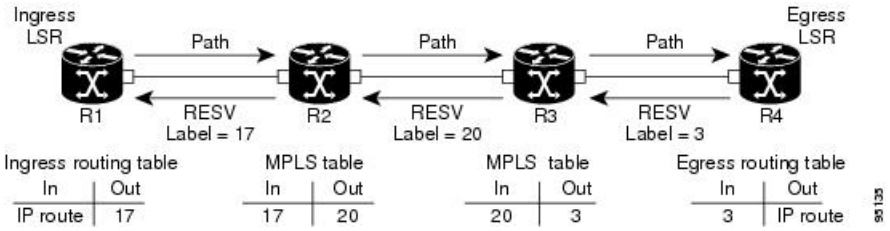
Related Topics

Configuring RSVP Packet Dropping, on page 61
Set DSCP for RSVP Packets: Example, on page 78
Verifying RSVP Configuration, on page 61

LSP Setup

LSP setup is initiated when the LSP head node sends path messages to the tail node (see the RSVP Operation figure).

Figure 6: RSVP Operation



The Path messages reserve resources along the path to each node, creating Path soft states 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. 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.

High Availability

RSVP is designed to ensure nonstop forwarding under the following constraints:

- Ability to tolerate the failure of one RP of a 1:1 redundant pair.
- Hitless software upgrade.

The RSVP high availability (HA) design follows the constraints of the underlying architecture where processes can fail without affecting the operation of other processes. A process failure of RSVP or any of its collaborators does not cause any traffic loss or cause established LSPs to go down. When RSVP restarts, it recovers its signaling states from its neighbors. No special configuration or manual intervention are required. You may configure RSVP graceful restart, which offers a standard mechanism to recover RSVP state information from neighbors after a failure.

Graceful Restart

RSVP graceful restart provides a control plane mechanism to ensure high availability (HA), which allows detection and recovery from failure conditions while preserving nonstop forwarding services on the systems running Cisco IOS XR software.

RSVP graceful restart provides a mechanism that minimizes the negative effects on MPLS traffic caused by these types of faults:

- Disruption of communication channels between two nodes when the communication channels are separate from the data channels. This is called *control channel failure*.
- Control plane of a node fails but the node preserves its data forwarding states. This is called *node failure*.

The procedure for RSVP graceful restart is described in the "Fault Handling" section of RFC 3473, *Generalized MPLS Signaling, RSVP-TE Extensions*. One of the main advantages of using RSVP graceful restart is recovery of the control plane while preserving nonstop forwarding and existing labels.

Note RSVP graceful restart feature is not supported when TE is running over multiple IGP instances which have different TE router-ids. This causes the TE tunnels to constantly flap.

Graceful Restart: Standard and Interface-Based

When you configure RSVP graceful restart, Cisco IOS XR software sends and expects node-id address based Hello messages (that is, Hello Request and Hello Ack messages). The RSVP graceful restart Hello session is not established if the neighbor router does not respond with a node-id based Hello Ack message.

You can also configure graceful restart to respond (send Hello Ack messages) to interface-address based Hello messages sent from a neighbor router in order to establish a graceful restart Hello session on the neighbor router. If the neighbor router does not respond with node-id based Hello Ack message, however, the RSVP graceful restart Hello session is not established.

Cisco IOS XR software provides two commands to configure graceful restart:

- **signalling hello graceful-restart**
- **signalling hello graceful-restart interface-based**

Note By default, graceful restart is disabled. To enable interface-based graceful restart, you must first enable standard graceful restart. You cannot enable interface-based graceful restart independently.

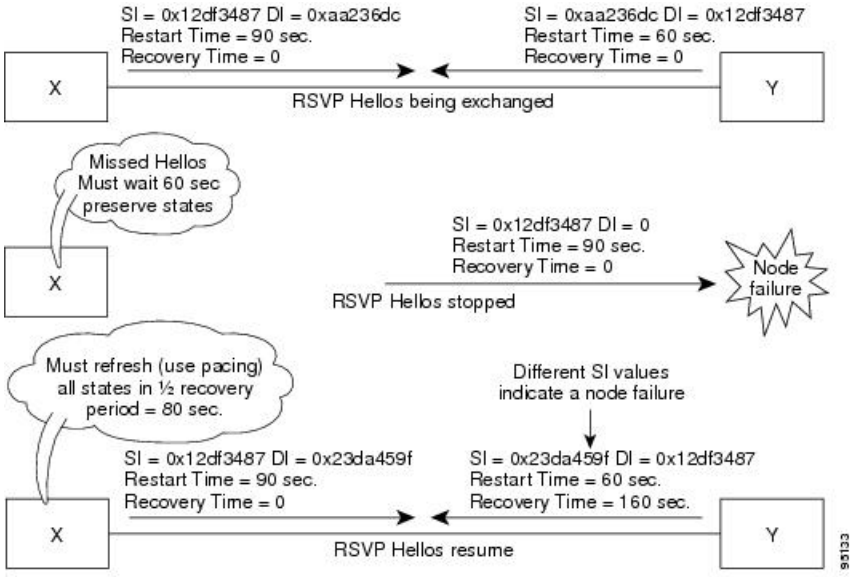
Related Topics

Enabling Graceful Restart, on page 59
Enable Graceful Restart: Example, on page 77
Enable Interface-Based Graceful Restart: Example, on page 77

Graceful Restart: Figure

Figure 7: Node Failure with RSVP

This figure illustrates how RSVP graceful restart handles a node failure condition.



RSVP graceful restart requires the use of RSVP hello messages. Hello messages are used between RSVP neighbors. Each neighbor 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. This means that a hello message contains either a hello Request or a hello ACK object. These two objects have the same format.

The restart cap object indicates a node's restart capabilities. It is carried in hello messages if the sending node supports state recovery. The restart cap object has the following two fields:

Restart Time

Time after a loss in Hello messages within which RSVP hello session can be reestablished. It is possible for a user to manually configure the Restart Time.

Recovery Time

Time that the sender waits for the recipient to re-synchronize states after the re-establishment of hello messages. This value is computed and advertised based on number of states that existed before the fault occurred.

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.

Restart cap objects are sent to an RSVP neighbor when RSVP states are shared with that neighbor. If 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 graceful restart is disabled, no hello messages (Requests or ACKs) are sent. If a hello Request message is received from an unknown neighbor, no hello ACK is sent back.

ACL-based Prefix Filtering

RSVP provides for the configuration of extended access lists (ACLs) to forward, drop, or perform normal processing on RSVP router-alert (RA) packets. Prefix filtering is designed for use at core access routers in order that RA packets (identified by a source/destination address) can be seamlessly forwarded across the core from one access point to another (or, conversely to be dropped at this node). RSVP applies prefix filtering rules only to RA packets because RA packets contain source and destination addresses of the RSVP flow.

Note RA packets forwarded due to prefix filtering must not be sent as RSVP bundle messages, because bundle messages are hop-by-hop and do not contain RA. Forwarding a Bundle message does not work, because the node receiving the messages is expected to apply prefix filtering rules only to RA packets.

For each incoming RSVP RA packet, RSVP inspects the IP header and attempts to match the source/destination IP addresses with a prefix configured in an extended ACL. The results are as follows:

- If an ACL does not exist, the packet is processed like a normal RSVP packet.
- If the ACL match yields an explicit permit (and if the packet is not locally destined), the packet is forwarded. The IP TTL is decremented on all forwarded packets.
- If the ACL match yields an explicit deny, the packet is dropped.

If there is no explicit permit or explicit deny, the ACL infrastructure returns an implicit (default) deny. RSVP can be configured to drop the packet. By default, RSVP processes the packet if the ACL match yields an implicit (default) deny.

Related Topics

Configuring ACLs for Prefix Filtering, on page 60
Configure ACL-based Prefix Filtering: Example, on page 77

RSVP MIB

RFC 2206, RSVP Management Information Base Using SMIv2 defines all the SNMP MIB objects that are relevant to RSVP. By implementing the RSVP MIB, you can perform these functions:

- Specifies two traps (NetFlow and LostFlow) which are triggered when a new flow is created or deleted.
- Lets you use SNMP to access objects belonging to RSVP.

Related Topics

Enabling RSVP Traps, on page 64
Enable RSVP Traps: Example, on page 78

Information About Implementing RSVP Authentication

Before implementing RSVP authentication, you must configure a keychain first. The name of the keychain must be the same as the one used in the keychain configuration. For more information about configuring keychains, see *System Security Configuration Guide for Cisco NCS 6000 Series Routers*.

Note RSVP authentication supports only keyed-hash message authentication code (HMAC) type algorithms.

To implement RSVP authentication on Cisco IOS XR software, you must understand the following concepts:

RSVP Authentication Functions

You can carry out these tasks with RSVP authentication:

- Set up a secure relationship with a neighbor by using secret keys that are known only to you and the neighbor.
- Configure RSVP authentication in global, interface, or neighbor configuration modes.
- Authenticate incoming messages by checking if there is a valid security relationship that is associated based on key identifier, incoming interface, sender address, and destination address.
- Add an integrity object with message digest to the outgoing message.
- Use sequence numbers in an integrity object to detect replay attacks.

RSVP Authentication Design

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. 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 following reasons 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.

Global configuration mode configures the defaults for interface and neighbor interface modes. These 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.

Related Topics

Configuring a Lifetime for an Interface for RSVP Authentication, on page 69
RSVP Authentication by Using All the Modes: Example, on page 80

Global, Interface, and Neighbor Authentication Modes

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.

Note RSVP uses the following rules when choosing which authentication parameter to use when that parameter is configured at multiple levels (interface, neighbor, or global). RSVP goes from the most specific to least specific; that is, neighbor, interface, and global.

Global keys simplify the configuration and eliminate the chances of a key mismatch when receiving messages from multiple neighbors and multiple interfaces. However, global keys do not provide the best security.

Interface keys are used to secure specific interfaces between two RSVP neighbors. Because many of the RSVP messages are IP routed, there are many scenarios in which using interface keys are not recommended. If all keys on the interfaces are not the same, there is a risk of a key mismatch for the following reasons:

- When the RSVP graceful restart is enabled, RSVP hello messages are sent with a source IP address of the local router ID and a destination IP address of the neighbor router ID. Because multiple routes can exist between the two neighbors, the RSVP hello message can traverse to different interfaces.
- When the RSVP fast reroute (FRR) is active, the RSVP Path and Resv messages can traverse multiple interfaces.
- When Generalized Multiprotocol Label Switching (GMPLS) optical tunnels are configured, RSVP messages are exchanged with router IDs as the source and destination IP addresses. Since multiple control channels can exist between the two neighbors, the RSVP messages can traverse different interfaces.

Neighbor-based keys are particularly useful in a network in which some neighbors support RSVP authentication procedures and others do not. When the neighbor-based keys are configured for a particular neighbor, you are advised to configure all the neighbor's addresses and router IDs for RSVP authentication.

Related Topics

Configuring a Lifetime for RSVP Authentication in Global Configuration Mode, on page 66
RSVP Authentication Global Configuration Mode: Example, on page 79
Specifying the RSVP Authentication Keychain in Interface Mode, on page 68
RSVP Authentication by Using All the Modes: Example, on page 80

Security Association

A security association (SA) is defined as a collection of information that is required to maintain secure communications with a peer to counter replay attacks, spoofing, and packet corruption.

This table lists the main parameters that define a security association.

Table 3: Security Association Main Parameters

Parameter	Description
src	IP address of the sender.
dst	IP address of the final destination.
interface	Interface of the SA.
direction	Send or receive type of the SA.
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.
digest	Algorithm last used (returned from the key-source).
Window Size	Specifies the tolerance for the configurable parameter. The parameter is applicable when the direction parameter is the receive type.
Window	Specifies the last *window size* value sequence number that is received or accepted. The parameter is applicable when the direction parameter is the receive type.

An SA is created dynamically when sending and receiving messages that require authentication. The neighbor, source, and destination addresses are obtained either from the IP header or from an RSVP object, such as a HOP object, and whether the message is incoming or outgoing.

When the SA is created, an expiration timer is created. When the SA authenticates a message, it is marked as recently used. The lifetime timer periodically checks if the SA is being used. If so, the flag is cleared and is cleaned up for the next period unless it is marked again.

This table shows how to locate the source and destination address keys for an SA that is based on the message type.

Table 4: Source and Destination Address Locations for Different Message Types

Message Type	Source Address Location	Destination Address Location
Path	HOP object	SESSION object
PathTear	HOP object	SESSION object
PathError	HOP object	IP header
Resv	HOP object	IP header
ResvTear	HOP object	IP header

Message Type	Source Address Location	Destination Address Location
ResvError	HOP object	IP header
ResvConfirm	IP header	CONFIRM object
Ack	IP header	IP header
Srefresh	IP header	IP header
Hello	IP header	IP header
Bundle	—	—

Related Topics

Specifying the Keychain for RSVP Neighbor Authentication, on page 71
RSVP Neighbor Authentication: Example, on page 80
Configuring a Lifetime for RSVP Neighbor Authentication, on page 72
RSVP Authentication Global Configuration Mode: Example, on page 79

Key-source Key-chain

The key-source key-chain is used to specify which keys to use.

You configure a list of keys with specific IDs and have different lifetimes so that keys are changed at predetermined intervals automatically, without any disruption of service. Rollover enhances network security by minimizing the problems that could result if an untrusted source obtained, deduced, or guessed the current key.

RSVP handles rollover by using the following key ID types:

- On TX, use the youngest eligible key ID.
- On RX, use the key ID that is received in an integrity object.

For more information about implementing keychain management, see *System Security Configuration Guide for Cisco NCS 6000 Series Routers*.

Related Topics

Enabling RSVP Authentication Using the Keychain in Global Configuration Mode, on page 66
RSVP Authentication Global Configuration Mode: Example, on page 79
Specifying the Keychain for RSVP Neighbor Authentication, on page 71
RSVP Neighbor Authentication: Example, on page 80

Guidelines for Window-Size and Out-of-Sequence Messages

These guidelines are required for window-size and out-of-sequence messages:

- Default window-size is set to 1. If a single message is received out-of-sequence, RSVP rejects it and displays a message.

- When RSVP messages are sent in burst mode (for example, tunnel optimization), some messages can become out-of-sequence for a short amount of time.
- Window size can be increased by using the **window-size** command. When the window size is increased, replay attacks can be detected with duplicate sequence numbers.

Related Topics

Configuring the Window Size for RSVP Authentication in Global Configuration Mode, on page 67
Configuring the Window Size for an Interface for RSVP Authentication, on page 70
Configuring the Window Size for RSVP Neighbor Authentication, on page 73
RSVP Authentication by Using All the Modes: Example, on page 80
RSVP Authentication for an Interface: Example, on page 79

Caveats for Out-of-Sequence

These caveats are listed for out-of-sequence:

- When RSVP messages traverse multiple interface types with different maximum transmission unit (MTU) values, some messages can become out-of-sequence if they are fragmented.
- Packets with some IP options may be reordered.
- Change in QoS configurations may lead to a transient reorder of packets.
- QoS policies can cause a reorder of packets in a steady state.

Because all out-of-sequence messages are dropped, the sender must retransmit them. Because RSVP state timeouts are generally long, out-of-sequence messages during a transient state do not lead to a state timeout.

How to Implement RSVP

RSVP requires coordination among several routers, establishing exchange of RSVP messages to set up LSPs. Depending on the client application, RSVP requires some basic configuration, as described in these topics:

Configuring Traffic Engineering Tunnel Bandwidth

To configure traffic engineering tunnel bandwidth, you must first set up TE tunnels and configure the reserved bandwidth per interface (there is no need to configure bandwidth for the data channel or the control channel).

Cisco IOS XR software supports two MPLS DS-TE modes: Prestandard and IETF.

Note For prestandard DS-TE you do not need to configure bandwidth for the data channel or the control channel. There is no other specific RSVP configuration required for this application. When no RSVP bandwidth is specified for a particular interface, you can specify zero bandwidth in the LSP setup if it is configured under RSVP interface configuration mode or MPLS-TE configuration mode.

Related Topics

Configuring a Prestandard DS-TE Tunnel, on page 117

Configuring an IETF DS-TE Tunnel Using RDM, on page 119
Configuring an IETF DS-TE Tunnel Using MAM, on page 121

Confirming DiffServ-TE Bandwidth

Perform this task to confirm DiffServ-TE bandwidth.

In RSVP global and subpools, reservable bandwidths are configured per interface to accommodate TE tunnels on the node. Available bandwidth from all configured bandwidth pools is advertised using IGP. RSVP signals the TE tunnel with appropriate bandwidth pool requirements.

SUMMARY STEPS

1. **configure**
2. **rsvp**
3. **interface** *type interface-path-id*
4. **bandwidth** *total-bandwidth max-flow* **sub-pool** *sub-pool-bw*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp** **Example:** `RP/0/RP0/CPU0:router(config)# rsvp`	Enters RSVP configuration mode.
Step 3	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-rsvp)# interface pos 0/2/0/0`	Enters interface configuration mode for the RSVP protocol.
Step 4	**bandwidth** *total-bandwidth max-flow* **sub-pool** *sub-pool-bw* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth 1000 100 sub-pool 150`	Sets the reservable bandwidth, the maximum RSVP bandwidth available for a flow and the sub-pool bandwidth on this interface.
Step 5	**commit**	

Related Topics

Differentiated Services Traffic Engineering, on page 90
Bandwidth Configuration (MAM): Example, on page 75

Bandwidth Configuration (RDM): Example, on page 75

Enabling Graceful Restart

Perform this task to enable graceful restart for implementations using both node-id and interface-based hellos.

RSVP graceful restart provides a control plane mechanism to ensure high availability, which allows detection and recovery from failure conditions while preserving nonstop forwarding services.

SUMMARY STEPS

1. **configure**
2. **rsvp**
3. **signalling graceful-restart**
4. **signalling graceful-restart interface-based**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp** **Example:** `RP/0/RP0/CPU0:router(config)# rsvp`	Enters the RSVP configuration mode.
Step 3	**signalling graceful-restart** **Example:** `RP/0/RP0/CPU0:router(config-rsvp)# signalling graceful-restart`	Enables the graceful restart process on the node.
Step 4	**signalling graceful-restart interface-based** **Example:** `RP/0/RP0/CPU0:router(config-rsvp)# signalling graceful-restart interface-based`	Enables interface-based graceful restart process on the node.
Step 5	**commit**	

Related Topics

Graceful Restart: Standard and Interface-Based, on page 50
Enable Graceful Restart: Example, on page 77
Enable Interface-Based Graceful Restart: Example, on page 77

Configuring ACL-based Prefix Filtering

Two procedures are provided to show how RSVP Prefix Filtering is associated:

- Configuring ACLs for Prefix Filtering, on page 60
- Configuring RSVP Packet Dropping, on page 61

Configuring ACLs for Prefix Filtering

Perform this task to configure an extended access list ACL that identifies the source and destination prefixes used for packet filtering.

Note The extended ACL needs to be configured separately using extended ACL configuration commands.

SUMMARY STEPS

1. **configure**
2. **rsvp**
3. **signalling prefix-filtering access-list**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp** **Example:** `RP/0/RP0/CPU0:router(config)# rsvp`	Enters the RSVP configuration mode.
Step 3	**signalling prefix-filtering access-list** **Example:** `RP/0/RP0/CPU0:router(config-rsvp)# signalling prefix-filtering access-list banks`	Enter an extended access list name as a string.
Step 4	**commit**	

Related Topics

ACL-based Prefix Filtering, on page 52
Configure ACL-based Prefix Filtering: Example, on page 77

Configuring RSVP Packet Dropping

Perform this task to configure RSVP to drop RA packets when the ACL match returns an implicit (default) deny.

The default behavior performs normal RSVP processing on RA packets when the ACL match returns an implicit (default) deny.

SUMMARY STEPS

1. **configure**
2. **rsvp**
3. **signalling prefix-filtering default-deny-action**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp** **Example:** `RP/0/RP0/CPU0:router(config)# rsvp`	Enters the RSVP configuration mode.
Step 3	**signalling prefix-filtering default-deny-action** **Example:** `RP/0/RP0/CPU0:router(config-rsvp)# signalling prefix-filtering default-deny-action`	Drops RA messages.
Step 4	**commit**	

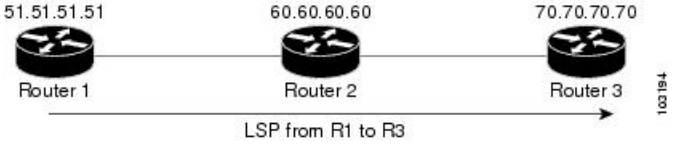
Related Topics

Overview of RSVP for MPLS-TE , on page 48
Set DSCP for RSVP Packets: Example, on page 78

Verifying RSVP Configuration

Figure 8: Sample Topology

This figure illustrates the topology.



Perform the following steps to verify RSVP configuration.

SUMMARY STEPS

1. **show rsvp session**
2. **show rsvp counters messages summary**
3. **show rsvp counters events**
4. **show rsvp interface** *type interface-path-id* [**detail**]
5. **show rsvp graceful-restart**
6. **show rsvp graceful-restart** [**neighbors** *ip-address* | **detail**]
7. **show rsvp interface**
8. **show rsvp neighbor**

DETAILED STEPS

Step 1 show rsvp session

Verifies that all routers on the path of the LSP are configured with at least one Path State Block (PSB) and one Reservation State Block (RSB) per session.

Example:

```
RP/0/RP0/CPU0:router# show rsvp session

Type Destination Add DPort Proto/ExtTunID PSBs RSBs Reqs
 ---- --------------- ----- --------------- ----- ----- ----- LSP4
 172.16.70.70 6 10.51.51.51 1 1 0
```

In the example , the output represents an LSP from ingress (head) router 10.51.51.51 to egress (tail) router 172.16.70.70. The tunnel ID (also called the *destination port*) is 6.

Example:

```
If no states can be found for a session that should be up, verify the
      application (for example, MPLS-TE ) to see if
      everything is in order. If a session has one PSB but no RSB, this indicates
      that either the Path message is not making it to the egress (tail) router or
      the reservation message is not making it back to the router R1 in question.
```

Go to the downstream router R2 and display the session information:

Example:

```
If R2 has no PSB, either the path message is not making it to the
      router or the path message is being rejected (for example, due to lack of
      resources). If R2 has a PSB but no RSB, go to the next downstream router R3
      to investigate. If R2 has a PSB and an RSB, this means the reservation is
      not making it from R2 to R1 or is being rejected.
```

Step 2 show rsvp counters messages summary

Verifies whether the RSVP message is being transmitted and received.

Example:

```
RP/0/RP0/CPU0:router# show rsvp counters messages summary
```

```
All RSVP Interfaces Recv Xmit Recv Xmit Path 0 25
      Resv 30 0 PathError 0 0 ResvError 0 1 PathTear 0 30 ResvTear 12 0
      ResvConfirm 0 0 Ack 24 37 Bundle 0 Hello 0 5099 SRefresh 8974 9012
      OutOfOrder 0 Retransmit 20 Rate Limited 0
```

Step 3 **show rsvp counters events**

Verifies how many RSVP states have expired. Because RSVP uses a soft-state mechanism, some failures will lead to RSVP states to expire due to lack of refresh from the neighbor.

Example:

```
RP/0/RP0/CPU0:router# show rsvp counters events

mgmtEthernet0/0/0/0 tunnel6 Expired Path states 0 Expired
      Path states 0 Expired Resv states 0 Expired Resv states 0 NACKs received 0
      NACKs received 0 POS0/3/0/0                              POS0/3/0/1 Expired
      Path states 0 Expired Path states 0 Expired Resv states 0 Expired Resv
      states 0 NACKs received 0 NACKs received 0 POS0/3/0/2
                                POS0/3/0/3 Expired Path states 0 Expired Path
      states 0 Expired Resv states 0 Expired Resv states 1 NACKs received 0 NACKs
      received 1
```

Step 4 **show rsvp interface** *type interface-path-id* [**detail**]

Verifies that refresh reduction is working on a particular interface.

Example:

```
RP/0/RP0/CPU0:router# show rsvp interface pos0/3/0/3 detail

INTERFACE: POS0/3/0/3 (ifh=0x4000D00). BW
      (bits/sec): Max=1000M. MaxFlow=1000M. Allocated=1K (0%). MaxSub=0.
      Signalling: No DSCP marking. No rate limiting. States in: 1. Max missed
      msgs: 4. Expiry timer: Running (every 30s). Refresh interval: 45s. Normal
      Refresh timer: Not running. Summary refresh timer: Running. Refresh
      reduction local: Enabled. Summary Refresh: Enabled (4096 bytes max).
      Reliable summary refresh: Disabled. Ack hold: 400 ms, Ack max size: 4096
      bytes. Retransmit: 900ms. Neighbor information: Neighbor-IP Nbor-MsgIds
      States-out Refresh-Reduction Expiry(min::sec) -------------- --------------
      ---------- ------------------ ---------------- 64.64.64.65 1 1 Enabled
      14::45
```

Step 5 **show rsvp graceful-restart**

Verifies that graceful restart is enabled locally.

Example:

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

Graceful restart: enabled Number of global
      neighbors: 1 Local MPLS router id: 10.51.51.51 Restart time: 60 seconds
      Recovery time: 0 seconds Recovery timer: Not running Hello interval: 5000
      milliseconds Maximum Hello miss-count: 3
```

Step 6 **show rsvp graceful-restart** [**neighbors** *ip-address* | **detail**]

Verifies that graceful restart is enabled on the neighbor(s). These examples show that neighbor 192.168.60.60 is not responding to hello messages.

Example:

```
RP/0/RP0/CPU0:router# show rsvp graceful-restart neighbors 192.168.60.60

Neighbor App State Recovery Reason
      Since LostCnt --------------- ----- ------ -------- ------------
      -------------------- -------- 192.168.60.60 MPLS INIT DONE N/A 12/06/2003
      19:01:49 0
RP/0/RP0/CPU0:router# show rsvp graceful-restart neighbors detail

Neighbor: 192.168.60.60 Source: 10.51.51.51
      (MPLS) Hello instance for application MPLS Hello State: INIT (for 3d23h)
      Number of times communications with neighbor lost: 0 Reason: N/A Recovery
      State: DONE Number of Interface neighbors: 1 address: 10.64.64.65 Restart
      time: 0 seconds Recovery time: 0 seconds Restart timer: Not running Recovery
      timer: Not running Hello interval: 5000 milliseconds Maximum allowed missed
      Hello messages: 3
```

Step 7 **show rsvp interface**

Verifies the available RSVP bandwidth.

Example:

```
RP/0/RP0/CPU0:router# show rsvp interface

Interface MaxBW MaxFlow Allocated MaxSub -----------
      -------- -------- --------------- -------- Et0/0/0/0 0 0 0 ( 0%) 0 PO0/3/0/0
      1000M 1000M 0 ( 0%) 0 PO0/3/0/1 1000M 1000M 0 ( 0%) 0 PO0/3/0/2 1000M 1000M
      0 ( 0%) 0 PO0/3/0/3 1000M 1000M 1K ( 0%) 0
```

Step 8 **show rsvp neighbor**

Verifies the RSVP neighbors.

Example:

```
RP/0/RP0/CPU0:router# show rsvp neighbor detail
Global Neighbor: 40.40.40.40 Interface Neighbor: 1.1.1.1
      Interface: POS0/0/0/0 Refresh Reduction: "Enabled" or "Disabled". Remote
      epoch: 0xXXXXXXXX Out of order messages: 0 Retransmitted messages: 0
      Interface Neighbor: 2.2.2.2 Interface: POS0/1/0/0 Refresh Reduction:
      "Enabled" or "Disabled". Remote epoch: 0xXXXXXXXX Out of order messages: 0
      Retransmitted messages: 0
```

Related Topics

Overview of RSVP for MPLS-TE , on page 48

Enabling RSVP Traps

With the exception of the RSVP MIB traps, no action is required to activate the MIBs. This MIB feature is automatically enabled when RSVP is turned on; however, RSVP traps must be enabled.

Perform this task to enable all RSVP MIB traps, NewFlow traps, and LostFlow traps.

SUMMARY STEPS

1. **configure**
2. **snmp-server traps rsvp lost-flow**
3. **snmp-server traps rsvp new-flow**
4. **snmp-server traps rsvp all**
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**snmp-server traps rsvp lost-flow** **Example:** `RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp lost-flow`	Sends RSVP notifications to enable RSVP LostFlow traps.
Step 3	**snmp-server traps rsvp new-flow** **Example:** `RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp new-flow`	Sends RSVP notifications to enable RSVP NewFlow traps.
Step 4	**snmp-server traps rsvp all** **Example:** `RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp all`	Sends RSVP notifications to enable all RSVP MIB traps.
Step 5	**commit**	

Related Topics

RSVP MIB, on page 52
Enable RSVP Traps: Example, on page 78

How to Implement RSVP Authentication

There are three types of RSVP authentication modes—global, interface, and neighbor. These topics describe how to implement RSVP authentication for each mode:

Configuring Global Configuration Mode RSVP Authentication

These tasks describe how to configure RSVP authentication in global configuration mode:

Enabling RSVP Authentication Using the Keychain in Global Configuration Mode

Perform this task to enable RSVP authentication for cryptographic authentication by specifying the keychain in global configuration mode.

Note You must configure a keychain before completing this task (see *System Security Configuration Guide for Cisco NCS 6000 Series Routers*).

SUMMARY STEPS

1. **configure**
2. **rsvp authentication**
3. **key-source key-chain** *key-chain-name*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**rsvp authentication**
	Example:
	`RP/0/RP0/CPU0:router(config)# rsvp authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-auth)#`
		Enters RSVP authentication configuration mode.

Step 3	**key-source key-chain** *key-chain-name* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-auth)# key-source key-chain mpls-keys`	Specifies the source of the key information to authenticate RSVP signaling messages. ***key-chain-name*** Name of the keychain. The maximum number of characters is 32.
Step 4	**commit**	

Related Topics

Key-source Key-chain, on page 56
RSVP Authentication Global Configuration Mode: Example, on page 79

Configuring a Lifetime for RSVP Authentication in Global Configuration Mode

Perform this task to configure a lifetime value for RSVP authentication in global configuration mode.

SUMMARY STEPS

1. **configure**
2. **rsvp authentication**
3. **life-time** *seconds*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**rsvp authentication**
	Example:
	`RP/0/RP0/CPU0:router(config)# rsvp authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-auth)#`
		Enters RSVP authentication configuration mode.

Step 3	**life-time** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-auth)# life-time 2000`	Controls how long RSVP maintains security associations with other trusted RSVP neighbors. ***seconds*** Length of time (in seconds) that RSVP maintains idle security associations with other trusted RSVP neighbors. Range is from 30 to 86400. The default value is 1800.
Step 4	**commit**	

Related Topics

Global, Interface, and Neighbor Authentication Modes, on page 54
RSVP Authentication Global Configuration Mode: Example, on page 79

Configuring the Window Size for RSVP Authentication in Global Configuration Mode

Perform this task to configure the window size for RSVP authentication in global configuration mode.

SUMMARY STEPS

1. **configure**
2. **rsvp authentication**
3. **window-size** *N*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose

Step 2	**rsvp authentication**
	Example:
	`RP/0/RP0/CPU0:router(config)# rsvp authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-auth)#`
		Enters RSVP authentication configuration mode.

Step 3	**window-size** *N* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-auth)# window-size 33`	Specifies the maximum number of RSVP authenticated messages that can be received out-of-sequence. ***N*** Size of the window to restrict out-of-sequence messages. The range is from 1 to 64. The default value is 1, in which case all out-of-sequence messages are dropped.
Step 4	**commit**	

Related Topics

Guidelines for Window-Size and Out-of-Sequence Messages, on page 56
RSVP Authentication by Using All the Modes: Example, on page 80
RSVP Authentication for an Interface: Example, on page 79

Configuring an Interface for RSVP Authentication

These tasks describe how to configure an interface for RSVP authentication:

Specifying the RSVP Authentication Keychain in Interface Mode

Perform this task to specify RSVP authentication keychain in interface mode.

You must configure a keychain first (see *System Security Configuration Guide for Cisco NCS 6000 Series Routers*).

SUMMARY STEPS

1. **configure**
2. **rsvp interface** *type interface-path-id*
3. **authentication**
4. **key-source key-chain** *key-chain-name*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose

Step 2	**rsvp interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config)# rsvp interface POS 0/2/1/0`
	`RP/0/RP0/CPU0:router(config-rsvp-if)#`
		Enters RSVP interface configuration mode.

Step 3	**authentication**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-if)# authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-if-auth)#`
		Enters RSVP authentication configuration mode.

Step 4	**key-source key-chain** *key-chain-name* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if-auth)# key-source key-chain mpls-keys`	Specifies the source of the key information to authenticate RSVP signaling messages. ***key-chain-name*** Name of the keychain. The maximum number of characters is 32.
Step 5	**commit**	

Related Topics

Global, Interface, and Neighbor Authentication Modes, on page 54
RSVP Authentication by Using All the Modes: Example, on page 80

Configuring a Lifetime for an Interface for RSVP Authentication

Perform this task to configure a lifetime for the security association for an interface.

SUMMARY STEPS

1. **configure**
2. **rsvp interface** *type interface-path-id*
3. **authentication**
4. **life-time** *seconds*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp interface** *type interface-path-id* **Example:**	Enters RSVP interface configuration mode.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# rsvp interface POS 0/2/1/0` `RP/0/RP0/CPU0:router(config-rsvp-if)#`	

Step 3	**authentication**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-if)# authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-if-auth)#`
		Enters RSVP authentication configuration mode.

Step 4	**life-time** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if-auth)# life-time 2000`	Controls how long RSVP maintains security associations with other trusted RSVP neighbors. ***seconds*** Length of time (in seconds) that RSVP maintains idle security associations with other trusted RSVP neighbors. Range is from 30 to 86400. The default value is 1800.
Step 5	**commit**	

Related Topics

RSVP Authentication Design, on page 53
RSVP Authentication by Using All the Modes: Example, on page 80

Configuring the Window Size for an Interface for RSVP Authentication

Perform this task to configure the window size for an interface for RSVP authentication to check the validity of the sequence number received.

SUMMARY STEPS

1. **configure**
2. **rsvp interface** *type interface-path-d*
3. **authentication**
4. **window-size** *N*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp interface** *type interface-path-d* **Example:**	Enters RSVP interface configuration mode.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# rsvp interface POS 0/2/1/0` `RP/0/RP0/CPU0:router(config-rsvp-if)#`	

Step 3	**authentication**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-if)# authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-if-auth)#`
		Enters RSVP interface authentication configuration mode.

Step 4	**window-size** *N* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if-auth)# window-size 33`	Specifies the maximum number of RSVP authenticated messages that can be received out-of-sequence. ***N*** Size of the window to restrict out-of-sequence messages. The range is from 1 to 64. The default value is 1, in which case all out-of-sequence messages are dropped.
Step 5	**commit**	

Related Topics

Guidelines for Window-Size and Out-of-Sequence Messages, on page 56
RSVP Authentication by Using All the Modes: Example, on page 80
RSVP Authentication for an Interface: Example, on page 79

Configuring RSVP Neighbor Authentication

These tasks describe how to configure the RSVP neighbor authentication:

- Specifying the Keychain for RSVP Neighbor Authentication, on page 71
- Configuring a Lifetime for RSVP Neighbor Authentication, on page 72
- Configuring the Window Size for RSVP Neighbor Authentication, on page 73

Specifying the Keychain for RSVP Neighbor Authentication

Perform this task to specify the keychain RSVP neighbor authentication.

You must configure a keychain first (see *System Security Configuration Guide for Cisco NCS 6000 Series Routers*).

SUMMARY STEPS

1. **configure**
2. **rsvp neighbor** *IP-address* **authentication**
3. **key-source key-chain** *key-chain-name*

4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**rsvp neighbor** *IP-address* **authentication**
	Example:
	`RP/0/RP0/CPU0:router(config)# rsvp neighbor 1.1.1.1 authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-nbor-auth)#`
		Enters neighbor authentication configuration mode. Use the **rsvp neighbor** command to activate RSVP cryptographic authentication for a neighbor.
		IP address
		IP address of the neighbor. A single IP address for a specific neighbor; usually one of the neighbor's physical or logical (loopback) interfaces.
		authentication
		Configures the RSVP authentication parameters.

Step 3	**key-source key-chain** *key-chain-name* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-nbor-auth)# key-source key-chain mpls-keys`	Specifies the source of the key information to authenticate RSVP signaling messages. ***key-chain-name*** Name of the keychain. The maximum number of characters is 32.
Step 4	**commit**	

Related Topics

Key-source Key-chain, on page 56
Security Association, on page 54
RSVP Neighbor Authentication: Example, on page 80

Configuring a Lifetime for RSVP Neighbor Authentication

Perform this task to configure a lifetime for security association for RSVP neighbor authentication mode.

SUMMARY STEPS

1. **configure**
2. **rsvp neighbor** *IP-address* **authentication**
3. **life-time** *seconds*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose

Step 2	**rsvp neighbor** *IP-address* **authentication**
	Example:
	`RP/0/RP0/CPU0:router(config)# rsvp neighbor 1.1.1.1 authentication`
	`RP/0/RP0/CPU0:router(config-rsvp-nbor-auth)#`
		Enters RSVP neighbor authentication configuration mode. Use the **rsvp neighbor** command to specify a neighbor under RSVP.
		IP address
		IP address of the neighbor. A single IP address for a specific neighbor; usually one of the neighbor's physical or logical (loopback) interfaces.
		authentication
		Configures the RSVP authentication parameters.

Step 3	**life-time** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-rsvp-nbor-auth)# life-time 2000`	Controls how long RSVP maintains security associations with other trusted RSVP neighbors. The argument specifies the ***seconds*** Length of time (in seconds) that RSVP maintains idle security associations with other trusted RSVP neighbors. Range is from 30 to 86400. The default value is 1800.
Step 4	**commit**	

Related Topics

Security Association, on page 54
RSVP Authentication Global Configuration Mode: Example, on page 79

Configuring the Window Size for RSVP Neighbor Authentication

Perform this task to configure the RSVP neighbor authentication window size to check the validity of the sequence number received.

SUMMARY STEPS

1. **configure**
2. **rsvp neighbor** *IP address* **authentication**
3. **window-size** *N*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp neighbor** *IP address* **authentication** **Example:** `RP/0/RP0/CPU0:router(config)# rsvp neighbor 1.1.1.1`	Enters RSVP neighbor authentication configuration mode. Use the **rsvp neighbor** command to specify a neighbor under RSVP.

	Command or Action	Purpose
	authentication `RP/0/RP0/CPU0:router(config-rsvp-nbor-auth)#`	***IP address*** IP address of the neighbor. A single IP address for a specific neighbor; usually one of the neighbor's physical or logical (loopback) interfaces. **authentication** Configures the RSVP authentication parameters.

Step 3	**window-size** *N*
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-nbor-auth)#`
	`window-size 33`
		Specifies the maximum number of RSVP authenticated messages that is received out-of-sequence.
		N
		Size of the window to restrict out-of-sequence messages. The range is from 1 to 64. The default value is 1, in which case all out-of-sequence messages are dropped.

Step 4	**commit**	

Related Topics

Guidelines for Window-Size and Out-of-Sequence Messages, on page 56
RSVP Authentication by Using All the Modes: Example, on page 80
RSVP Authentication for an Interface: Example, on page 79

Verifying the Details of the RSVP Authentication

To display the security associations that RSVP has established with other RSVP neighbors, use the **show rsvp authentication** command.

Eliminating Security Associations for RSVP Authentication

To eliminate RSVP authentication SA's, use the **clear rsvp authentication** command. To eliminate RSVP counters for each SA, use the **clear rsvp counters authentication** command.

Configuration Examples for RSVP

Sample RSVP configurations are provided for some of the supported RSVP features.

- #unique_122
- #unique_123
- #unique_124
- Refresh Reduction and Reliable Messaging Configuration: Examples, on page 75
- Configure Graceful Restart: Examples, on page 77
- Configure ACL-based Prefix Filtering: Example, on page 77

- Set DSCP for RSVP Packets: Example, on page 78
- Enable RSVP Traps: Example, on page 78

Bandwidth Configuration (Prestandard): Example

The example shows the configuration of bandwidth on an interface using prestandard DS-TE mode. The example configures an interface for a reservable bandwidth of 7500, specifies the maximum bandwidth for one flow to be 1000 and adds a sub-pool bandwidth of 2000.

```
rsvp interface hundredGigE 0/0/0/0
 bandwidth 7500 1000 sub-pool 2000
```

Bandwidth Configuration (MAM): Example

The example shows the configuration of bandwidth on an interface using MAM. The example shows how to limit the total of all RSVP reservations on the hundredGigE 0/0/0/0 interface to 7500 kbps, and allow each single flow to reserve no more than 1000 kbps.

```
rsvp interface hundredGigE 0/0/0/0
 bandwidth mam 7500 1000
```

Related Topics

Confirming DiffServ-TE Bandwidth, on page 58
Differentiated Services Traffic Engineering, on page 90

Bandwidth Configuration (RDM): Example

The example shows the configuration of bandwidth on an interface using RDM. The example shows how to limit the total of all RSVP reservations on the hundredGigE 0/0/0/0 interface to 7500 kbps, and allows each single flow to reserve no more than 1000 kbps.

```
rsvp interface hundredGigE 0/0/0/0
 bandwidth rdm 7500 1000
```

Related Topics

Confirming DiffServ-TE Bandwidth, on page 58
Differentiated Services Traffic Engineering, on page 90

Refresh Reduction and Reliable Messaging Configuration: Examples

Refresh reduction feature as defined by RFC 2961 is supported and enabled by default. The examples illustrate the configuration for the refresh reduction feature. Refresh reduction is used with a neighbor only if the neighbor supports it also.

Refresh Interval and the Number of Refresh Messages Configuration: Example

The example shows how to configure the refresh interval to 30 seconds on POS 0/3/0/0 and how to change the number of refresh messages the node can miss before cleaning up the state from the default value of 4 to 6.

```
rsvp interface pos 0/3/0/0
 signalling refresh interval 30
 signalling refresh missed 6
```

Retransmit Time Used in Reliable Messaging Configuration: Example

The example shows how to set the retransmit timer to 2 seconds. To prevent unnecessary retransmits, the retransmit time value configured on the interface must be greater than the ACK hold time on its peer.

```
rsvp interface pos 0/4/0/1
 signalling refresh reduction reliable retransmit-time 2000
```

Acknowledgement Times Configuration: Example

The example shows how to change the acknowledge hold time from the default value of 400 ms, to delay or speed up sending of ACKs, and the maximum acknowledgment message size from default size of 4096 bytes. The example shows how to change the acknowledge hold time from the default value of 400 ms and how to delay or speed up sending of ACKs. The maximum acknowledgment message default size is from 4096 bytes.

```
rsvp interface pos 0/4/0/1
 signalling refresh reduction reliable ack-hold-time 1000
rsvp interface pos 0/4/0/1
 signalling refresh reduction reliable ack-max-size 1000
```

Note Ensure retransmit time on the peers' interface is at least twice the amount of the ACK hold time to prevent unnecessary retransmissions.

Summary Refresh Message Size Configuration: Example

The example shows how to set the summary refresh message maximum size to 1500 bytes.

```
rsvp interface pos 0/4/0/1
 signalling refresh reduction summary max-size 1500
```

Disable Refresh Reduction: Example

If the peer node does not support refresh reduction, or for any other reason you want to disable refresh reduction on an interface, the example shows how to disable refresh reduction on that interface.

```
rsvp interface pos 0/4/0/1
 signalling refresh reduction disable
```

Configure Graceful Restart: Examples

RSVP graceful restart is configured globally or per interface (as are refresh-related parameters). These examples show how to enable graceful restart, set the restart time, and change the hello message interval.

Enable Graceful Restart: Example

The example shows how to enable the RSVP graceful restart by default. If disabled, enable it with the following command.

```
rsvp signalling graceful-restart
```

Related Topics

Enabling Graceful Restart, on page 59
Graceful Restart: Standard and Interface-Based, on page 50

Enable Interface-Based Graceful Restart: Example

The example shows how to enable the RSVP graceful restart feature on an interface.

```
RP/0/RP0/CPU0:router#configure
RP/0/RP0/CPU0:router(config-rsvp)#interface bundle-ether 17
RP/0/RP0/CPU0:router(config-rsvp-if)#signalling hello graceful-restart ?
  interface-based  Configure Interface-based Hello
RP/0/RP0/CPU0:router(config-rsvp-if)#signalling hello graceful-restart interface-based
RP/0/RP0/CPU0:router(config-rsvp-if)#
```

Related Topics

Enabling Graceful Restart, on page 59
Graceful Restart: Standard and Interface-Based, on page 50

Change the Restart-Time: Example

The example shows how to change the restart time that is advertised in hello messages sent to neighbor nodes.

```
rsvp signalling graceful-restart restart-time 200
```

Change the Hello Interval: Example

The example shows how to change the interval at which RSVP graceful restart hello messages are sent per neighbor, and change the number of hellos missed before the neighbor is declared down.

```
rsvp signalling hello graceful-restart refresh interval 4000
rsvp signalling hello graceful-restart refresh misses 4
```

Configure ACL-based Prefix Filtering: Example

The example shows when RSVP receives a Router Alert (RA) packet from source address 1.1.1.1 and 1.1.1.1 is not a local address. The packet is forwarded with IP TTL decremented. Packets destined to 2.2.2.2 are dropped. All other RA packets are processed as normal RSVP packets.

```
show run ipv4 access-list
  ipv4 access-list rsvpacl
  10 permit ip host 1.1.1.1 any
```

```
    20 deny ip any host 2.2.2.2
    !
  show run rsvp
    rsvp
    signalling prefix-filtering access-list rsvpacl
    !
```

Related Topics

Configuring ACLs for Prefix Filtering, on page 60
ACL-based Prefix Filtering, on page 52

Set DSCP for RSVP Packets: Example

The configuration example sets the Differentiated Services Code Point (DSCP) field in the IP header of RSVP packets.

```
rsvp interface pos0/2/0/1
 signalling dscp 20
```

Related Topics

Configuring RSVP Packet Dropping, on page 61
Overview of RSVP for MPLS-TE , on page 48

Enable RSVP Traps: Example

The example enables the router to send all RSVP traps:

```
configure
 snmp-server traps rsvp all
```

The example enables the router to send RSVP LostFlow traps:

```
configure
 snmp-server traps rsvp lost-flow
```

The example enables the router to send RSVP RSVP NewFlow traps:

```
configure
 snmp-server traps rsvp new-flow
```

Related Topics

Enabling RSVP Traps, on page 64
RSVP MIB, on page 52

Configuration Examples for RSVP Authentication

These configuration examples are used for RSVP authentication:

- RSVP Authentication Global Configuration Mode: Example, on page 79

RSVP Authentication Global Configuration Mode: Example

The configuration example enables authentication of all RSVP messages and increases the default lifetime of the SAs.

```
rsvp
 authentication
  key-source key-chain default_keys
  life-time 3600
 !
!
```

Note The specified keychain (default_keys) must exist and contain valid keys, or signaling will fail.

Related Topics

Enabling RSVP Authentication Using the Keychain in Global Configuration Mode, on page 66
Key-source Key-chain, on page 56
Configuring a Lifetime for RSVP Authentication in Global Configuration Mode, on page 66
Global, Interface, and Neighbor Authentication Modes, on page 54
Configuring a Lifetime for RSVP Neighbor Authentication, on page 72
Security Association, on page 54

RSVP Authentication for an Interface: Example

The configuration example enables authentication of all RSVP messages that are being sent or received on one interface only, and sets the window-size of the SAs.

```
rsvp
 interface GigabitEthernet0/6/0/0
  authentication
   window-size 64
  !
 !
```

Note Because the key-source keychain configuration is not specified, the global authentication mode keychain is used and inherited. The global keychain must exist and contain valid keys or signaling fails.

Related Topics

Configuring the Window Size for RSVP Authentication in Global Configuration Mode, on page 67

Configuring the Window Size for an Interface for RSVP Authentication, on page 70
Configuring the Window Size for RSVP Neighbor Authentication, on page 73
Guidelines for Window-Size and Out-of-Sequence Messages, on page 56

RSVP Neighbor Authentication: Example

The configuration example enables authentication of all RSVP messages that are being sent to and received from only a particular IP address.

```
rsvp
 neighbor 10.0.0.1
  authentication
   key-source key-chain nbr_keys
  !
 !
!
```

Related Topics

Specifying the Keychain for RSVP Neighbor Authentication, on page 71
Key-source Key-chain, on page 56
Security Association, on page 54

RSVP Authentication by Using All the Modes: Example

The configuration example shows how to perform the following functions:

- Authenticates all RSVP messages.
- Authenticates the RSVP messages to or from 10.0.0.1 by setting the keychain for the **key-source key-chain** command to nbr_keys, SA lifetime is set to 3600, and the default window-size is set to 1.
- Authenticates the RSVP messages not to or from 10.0.0.1 by setting the keychain for the **key-source key-chain** command to default_keys, SA lifetime is set to 3600, and the window-size is set 64 when using GigabitEthernet0/6/0/0; otherwise, the default value of 1 is used.

```
rsvp
 interface GigabitEthernet0/6/0/0
  authentication
   window-size 64
  !
 !
 neighbor 10.0.0.1
  authentication
   key-source key-chain nbr_keys
  !
 !
 authentication
  key-source key-chain default_keys
  life-time 3600
 !
!
```

Note If a keychain does not exist or contain valid keys, this is considered a configuration error because signaling fails. However, this can be intended to prevent signaling. For example, when using the above configuration, if the nbr_keys does not contain valid keys, all signaling with 10.0.0.1 fails.

Related Topics

Configuring the Window Size for RSVP Authentication in Global Configuration Mode, on page 67
Configuring the Window Size for an Interface for RSVP Authentication, on page 70
Configuring the Window Size for RSVP Neighbor Authentication, on page 73
Guidelines for Window-Size and Out-of-Sequence Messages, on page 56
Specifying the RSVP Authentication Keychain in Interface Mode, on page 68
Global, Interface, and Neighbor Authentication Modes, on page 54
Configuring a Lifetime for an Interface for RSVP Authentication, on page 69
RSVP Authentication Design, on page 53

Additional References

For additional information related to implementing GMPLS UNI, refer to the following references:

Related Documents

Related Topic	Document Title
GMPLS UNI commands	*GMPLS UNI Commands* module in *MPLS Command Reference for Cisco NCS 6000 Series Routers*
MPLS Traffic Engineering commands	*MPLS Traffic Engineering commands* module in *MPLS Command Reference for Cisco NCS 6000 Series Routers*
RSVP commands	*RSVP commands* module in *MPLS Command Reference for Cisco NCS 6000 Series Routers*
Getting started material	
Information about user groups and task IDs	*Configuring AAA Services* module in *System Security Configuration Guide for Cisco NCS 6000 Series Routers*

MIBs

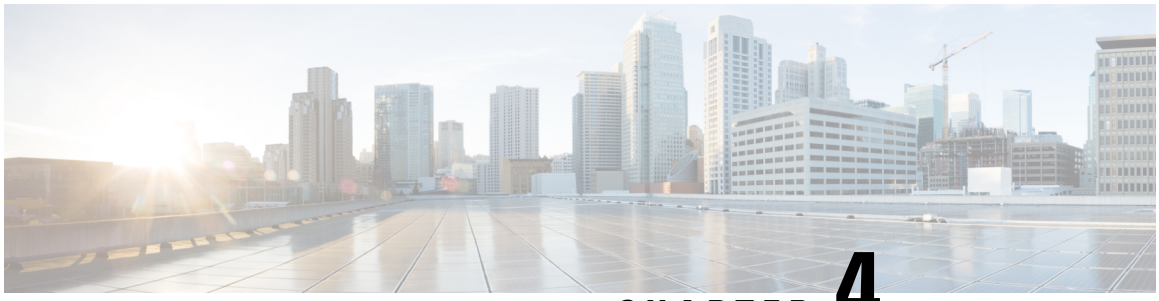
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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



CHAPTER 4

Implementing MPLS Forwarding

All Multiprotocol Label Switching (MPLS) features require a core set of MPLS label management and forwarding services; the MPLS Forwarding Infrastructure (MFI) supplies these services.

Feature History for Implementing MPLS-TE

Release	Modification
Release 5.0.0	This feature was introduced.

Prerequisites for Implementing Cisco MPLS Forwarding

These prerequisites are required to implement MPLS Forwarding:

- 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.
- Installed composite mini-image and the MPLS package, or a full composite image.

Restrictions for Implementing Cisco MPLS Forwarding

- Label switching on a Cisco router requires that Cisco Express Forwarding (CEF) be enabled.
- CEF is mandatory for Cisco IOS XR software and it does not need to be enabled explicitly.

Information About Implementing MPLS Forwarding

To implement MPLS Forwarding, you should understand these concepts:

MPLS Forwarding Overview

MPLS combines the performance and capabilities of Layer 2 (data link layer) switching with the proven scalability of Layer 3 (network layer) routing. MPLS enables service providers to meet the challenges of growth in network utilization while providing the opportunity to differentiate services without sacrificing the existing network infrastructure. The MPLS architecture is flexible and can be employed in any combination of Layer 2 technologies. MPLS support is offered for all Layer 3 protocols, and scaling is possible well beyond that typically offered in today's networks.

Based on routing information that is stored in the VRF IP routing table and VRF CEF table, packets are forwarded to their destination using MPLS.

A PE router binds a label to each customer prefix learned from a CE router and includes the label in the network reachability information for the prefix that it advertises to other PE routers. When a PE router forwards a packet received from a CE router across the provider network, it labels the packet with the label learned from the destination PE router. When the destination PE router receives the labeled packet it pops the label and uses it to direct the packet to the correct CE router. Label forwarding across the provider backbone, is based on either dynamic label switching or traffic engineered paths. A customer data packet carries two levels of labels when traversing the backbone:

- Top label directs the packet to the correct PE router
- Second label indicates how that PE router should forward the packet to the CE router

Label Switching Functions

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

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

In label switching, the analysis of the Layer 3 header is done only once. The Layer 3 header is then mapped into a fixed-length, unstructured value called a *label*.

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

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

Once a label is assigned, a short label header is added at the front of the Layer 3 packet. This header is carried across the network as part of the packet. At subsequent hops through each MPLS router in the network, labels are swapped and forwarding decisions are made by means of MPLS forwarding table lookup for the label

carried in the packet header. Hence, the packet header does not need to be reevaluated during packet transit through the network. Because the label is of fixed length and unstructured, the MPLS forwarding table lookup process is both straightforward and fast.

Distribution of Label Bindings

Each label switching router (LSR) in the network makes an independent, local decision as to which label value to use to represent a forwarding equivalence class. This association is known as a label binding.

Note The distribution of label bindings cannot be done statically for the Layer 2 VPN pseudowire.

Each LSR informs its neighbors of the label bindings it has made. This awareness of label bindings by neighboring routers is facilitated by these protocols:

Label Distribution Protocol (LDP)

Supports MPLS forwarding along normally routed paths.

Resource Reservation Protocol (RSVP)

Supports MPLS traffic engineering.

Border Gateway Protocol (BGP)

Supports MPLS virtual private networks (VPNs).

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

MFI Control-Plane Services

The MFI control-plane provides services to MPLS applications, such as Label Distribution Protocol (LDP) and Traffic Engineering (TE), that include enabling and disabling MPLS on an interface, local label allocation, MPLS rewrite setup (including backup links), management of MPLS label tables, and the interaction with other forwarding paths (IP Version 4 [IPv4] for example) to set up imposition and disposition.

MFI Data-Plane Services

The MFI data-plane provides a software implementation of MPLS forwarding in all of these forms:

- Imposition
- Disposition
- Label swapping

MPLS Maximum Transmission Unit

MPLS maximum transmission unit (MTU) indicates that the maximum size of the IP packet can still be sent on a data link, without fragmenting the packet. In addition, data links in MPLS networks have a specific MTU,

but for labeled packets. All IPv4 packets have one or more labels. This does imply that the labeled packets are slightly bigger than the IP packets, because for every label, four bytes are added to the packet. So, if n is the number of labels, n * 4 bytes are added to the size of the packet when the packet is labeled. The MPLS MTU parameter pertains to labeled packets.

How to Implement MPLS Forwarding

These topics explain how to configure a router for MPLS forwarding.

Additional References

For additional information related to implementing MPLS Forwarding, refer to the following references:

Related Documents

Standards

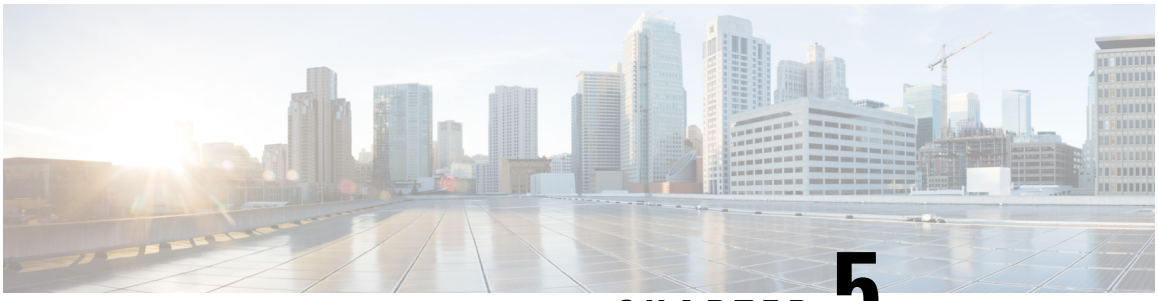
Standards	Title
	—

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 3031	*Multiprotocol Label Switching Architecture*
RFC 3443	*Time to Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks*
RFC 4105	*Requirements for Inter-Area MPLS Traffic Engineering*



CHAPTER 5

Implementing MPLS Traffic Engineering

Multiprotocol Label Switching (MPLS) is a standards-based solution driven by the Internet Engineering Task Force (IETF) that was devised to convert the Internet and IP backbones from best-effort networks into business-class transport mediums.

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) software enables an MPLS backbone to replicate and expand upon the TE capabilities of Layer 2 ATM and Frame Relay networks. MPLS is an integration of Layer 2 and Layer 3 technologies. By making traditional Layer 2 features available to Layer 3, MPLS enables traffic engineering. Thus, you can offer in a one-tier network what now can be achieved only by overlaying a Layer 3 network on a Layer 2 network.

Note The LMP and GMPLS-NNI features are not supported on PRP hardware.

Feature History for Implementing MPLS-TE

Release	Modification
Release 5.0.0	This feature was introduced.
Release 5.2.1	Support was added for these features: • Point-to-Multipoint Traffic-Engineering • Policy-Based Tunnel Selection
Release 5.2.5	Interarea P2MP Path Expansion within a Domain feature was added.
Release 6.1.2	Named Tunnel feature was added.
Release 6.4.1	Enabling Forward Class Zero in PBTS feature was added.

Prerequisites for Implementing Cisco MPLS Traffic Engineering

These prerequisites are required to implement MPLS TE:

- 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 .
- Installed composite mini-image and the MPLS package, or a full composite image.
- IGP activated.
- To configure Point-to-Multipoint (P2MP)-TE, a base set of RSVP and TE configuration parameters on ingress, midpoint, and egress nodes in the MPLS network is required. In addition, Point-to-Point (P2P) parameters are required.

Information About Implementing MPLS Traffic Engineering

To implement MPLS-TE, you should understand these concepts:

Overview of MPLS Traffic Engineering

MPLS-TE software enables an MPLS backbone to replicate and expand upon the traffic engineering capabilities of Layer 2 ATM and Frame Relay networks. MPLS is an integration of Layer 2 and Layer 3 technologies. By making traditional Layer 2 features available to Layer 3, MPLS enables traffic engineering. Thus, you can offer in a one-tier network what now can be achieved only by overlaying a Layer 3 network on a Layer 2 network.

MPLS-TE is essential for service provider and Internet service provider (ISP) backbones. Such backbones must support a high use of transmission capacity, and the networks must be very resilient so that they can withstand link or node failures. MPLS-TE provides an integrated approach to traffic engineering. With MPLS, traffic engineering capabilities are integrated into Layer 3, which optimizes the routing of IP traffic, given the constraints imposed by backbone capacity and topology.

Related Topics

Configuring Forwarding over the MPLS-TE Tunnel , on page 112

Benefits of MPLS Traffic Engineering

MPLS-TE enables ISPs to route network traffic to offer the best service to their users in terms of throughput and delay. By making the service provider more efficient, traffic engineering reduces the cost of the network.

Currently, some ISPs base their services on an overlay model. In the overlay model, transmission facilities are managed by Layer 2 switching. The routers see only a fully meshed virtual topology, making most destinations appear one hop away. If you use the explicit Layer 2 transit layer, you can precisely control how traffic uses available bandwidth. However, the overlay model has numerous disadvantages. MPLS-TE achieves the TE benefits of the overlay model without running a separate network and without a non-scalable, full mesh of router interconnects.

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 means of 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. MPLS-TE is built on these mechanisms:

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 performs bookkeeping on topology and resource information to be flooded.

Link-state IGP (Intermediate System-to-Intermediate System [IS-IS] or Open Shortest Path First [OSPF]—each with traffic engineering extensions)

These IGPs are used to globally flood topology and resource information from the link management module.

Enhancements to the shortest path first (SPF) calculation used by the link-state IGP (IS-IS or OSPF)

The IGP automatically routes traffic to the appropriate LSP tunnel, based on tunnel destination. Static routes can also be used to direct traffic to LSP tunnels.

Label switching forwarding

This forwarding mechanism provides routers with a Layer 2-like ability to direct traffic across multiple hops of the LSP established by RSVP signaling.

One approach to engineering a backbone is to define a mesh of tunnels from every ingress device to every egress device. The MPLS-TE path calculation and signaling modules determine the path taken by the LSPs for these tunnels, subject to resource availability and the dynamic state of the network.

The IGP (operating at an ingress device) determines which traffic should go to which egress device, and steers that traffic into the tunnel from ingress to egress. A flow from an ingress device to an egress device might be so large that it cannot fit over a single link, so it cannot be carried by a single tunnel. In this case, multiple tunnels between a given ingress and egress can be configured, and the flow is distributed using load sharing among the tunnels.

Note GRE over MPLS-TE tunnel is not supported. Hence, you cannot carry GRE traffic over an LSP established for MPLS-TE tunnel using RSVP-TE. This restriction also applies to SR-TE tunnels.

Related Topics

Building MPLS-TE Topology, on page 108
Creating an MPLS-TE Tunnel, on page 110
Build MPLS-TE Topology and Tunnels: Example, on page 151

Protocol-Based CLI

Cisco IOS XR software provides a protocol-based command line interface. The CLI provides commands that can be used with the multiple IGP protocols supported by MPLS-TE.

Differentiated Services Traffic Engineering

MPLS Differentiated Services (Diff-Serv) 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), users can configure multiple bandwidth constraints. These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint.

MPLS DS-TE provides the ability to configure multiple bandwidth constraints on an MPLS-enabled interface. Available bandwidths from all configured bandwidth constraints are advertised using IGP. TE tunnel is configured with bandwidth value and class-type requirements. Path calculation and admission control take the bandwidth and class-type into consideration. RSVP is used to signal the TE tunnel with bandwidth and class-type requirements.

MPLS DS-TE is deployed with either Russian Doll Model (RDM) or Maximum Allocation Model (MAM) for bandwidth calculations.

Cisco IOS XR software supports two DS-TE modes: Prestandard and IETF.

Related Topics

Confirming DiffServ-TE Bandwidth, on page 58
Bandwidth Configuration (MAM): Example, on page 75
Bandwidth Configuration (RDM): Example, on page 75

Prestandard DS-TE Mode

Prestandard 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. Note that prestandard DS-TE is enabled only after configuring the sub-pool bandwidth values on MPLS-enabled interfaces.

Prestandard Diff-Serve 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 Prestandard DS-TE mode.

Related Topics

Configuring a Prestandard DS-TE Tunnel, on page 117
Configure IETF DS-TE Tunnels: Example, on page 152

IETF DS-TE Mode

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 RDM and 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.

Bandwidth Constraint Models

IETF DS-TE mode provides support for the RDM and MAM bandwidth constraints models. Both models support up to two bandwidth pools.

Cisco IOS XR software provides global configuration for the switching between bandwidth constraint models. Both models can be configured on a single interface to preconfigure the bandwidth constraints before swapping to an alternate bandwidth constraint model.

Note NSF is not guaranteed when you change the bandwidth constraint model or configuration information.

By default, RDM is the default bandwidth constraint model used in both pre-standard and IETF mode.

Maximum Allocation Bandwidth Constraint Model

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.

Related Topics

Configuring an IETF DS-TE Tunnel Using MAM, on page 121

Russian Doll Bandwidth Constraint Model

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 in conjunction 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.

Note We recommend that RDM not be used in DS-TE environments in which the use of preemption is precluded. Although RDM ensures bandwidth efficiency and protection against QoS degradation of class types, it does guarantee isolation across class types.

Related Topics

Configuring an IETF DS-TE Tunnel Using RDM, on page 119

TE Class Mapping

Each of the eight available bandwidth values advertised in the IGP corresponds to a TE class. Because the IGP advertises only eight bandwidth values, there can be a maximum of only eight TE classes supported in an IETF DS-TE network.

TE class mapping must be exactly the same on all routers in a DS-TE domain. It is the responsibility of the operator configure these settings properly as there is no way to automatically check or enforce consistency.

The operator must configure TE tunnel class types and priority levels to form a valid TE class. When the TE class map configuration is changed, tunnels already up are brought down. Tunnels in the down state, can be set up if a valid TE class map is found.

The default TE class and attributes are listed. The default mapping includes four class types.

Table 5: TE Classes and Priority

TE Class	Class Type	Priority
0	0	7
1	1	7
2	Unused	—
3	Unused	—
4	0	0
5	1	0
6	Unused	—
7	Unused	—

Flooding

Available bandwidth in all configured bandwidth pools is flooded on the network to calculate accurate constraint paths when a new TE tunnel is configured. Flooding uses IGP protocol extensions and mechanisms to determine when to flood the network with bandwidth.

Flooding Triggers

TE Link Management (TE-Link) notifies IGP for both global pool and sub-pool available bandwidth and maximum bandwidth to flood the network in these events:

- Periodic timer expires (this does not depend on bandwidth pool type).
- Tunnel origination node has out-of-date information for either available global pool or sub-pool bandwidth, causing tunnel admission failure at the midpoint.
- Consumed bandwidth crosses user-configured thresholds. The same threshold is used for both global pool and sub-pool. If one bandwidth crosses the threshold, both bandwidths are flooded.

Flooding Thresholds

Flooding frequently can burden a network because all routers must send out and process these updates. Infrequent flooding causes tunnel heads (tunnel-originating nodes) to have out-of-date information, causing tunnel admission to fail at the midpoints.

You can control the frequency of flooding by configuring a set of thresholds. When locked bandwidth (at one or more priority levels) crosses one of these thresholds, flooding is triggered.

Thresholds apply to a percentage of the maximum available bandwidth (the global pool), which is locked, and the percentage of maximum available guaranteed bandwidth (the sub-pool), which is locked. If, for one or more priority levels, either of these percentages crosses a threshold, flooding is triggered.

Note Setting up a global pool TE tunnel can cause the locked bandwidth allocated to sub-pool tunnels to be reduced (and hence to cross a threshold). A sub-pool TE tunnel setup can similarly cause the locked bandwidth for global pool TE tunnels to cross a threshold. Thus, sub-pool TE and global pool TE tunnels can affect each other when flooding is triggered by thresholds.

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.

FRR (link or node) is supported over sub-pool tunnels the same way as for regular TE tunnels. In particular, when link protection is activated for a given link, TE tunnels eligible for FRR are redirected into the protection LSP, regardless of whether they are sub-pool or global pool tunnels.

Note The ability to configure FRR on a per-LSP basis makes it possible to provide different levels of fast restoration to tunnels from different bandwidth pools.

You should be aware of these requirements for the backup tunnel path:

- Backup tunnel must not pass through the element it protects.
- Primary tunnel and a backup tunnel should intersect at least at two points (nodes) on the path: point of local repair (PLR) and merge point (MP). PLR is the headend of the backup tunnel, and MP is the tailend of the backup tunnel.

Note When you configure TE tunnel with multiple protection on its path and merge point is the same node for more than one protection, you must configure record-route for that tunnel.

Note If FRR is greater than 50ms, it might lead to a loss of traffic.

Related Topics

Protecting MPLS Tunnels with Fast Reroute, on page 114

MPLS-TE and Fast Reroute over Link Bundles

These link bundle types are supported for MPLS-TE/FRR:

- Over Ethernet link bundles.
- Over VLANs over Ethernet link bundles.
- Number of links are limited to 100 for MPLS-TE and FRR.
- VLANs go over any Ethernet interface (for example,).

FRR is supported over bundle interfaces in the following ways:

- Uses minimum links as a threshold to trigger FRR over a bundle interface.
- Uses the minimum total available bandwidth as a threshold to trigger FRR.

Ignore Intermediate System-to-Intermediate System Overload Bit Setting in MPLS-TE

The Ignore Intermediate System-to-Intermediate System (IS-IS) Overload Bit Setting in MPLS-TE feature ensures that the RSVP-TE LSPs are not broken because of routers that enabled the IS-IS overload bit.

Note The current implementation does not allow nodes that have indicated an overload situation through the IS-IS overload bit.

Therefore, an overloaded node cannot be used. The IS-IS overload bit limitation is an indication of an overload situation in the IP topology. The feature provides a method to prevent an IS-IS overload condition from affecting MPLS-TE.

Enhancement Options of IS-IS OLA

Related Topics

Configuring the Ignore Integrated IS-IS Overload Bit Setting in MPLS-TE, on page 125
Configure the Ignore IS-IS Overload Bit Setting in MPLS-TE: Example, on page 153

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 MPLS-TE tunnels.

In the traditional TE scheme, 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 in the command-line interface (CLI). Furthermore, you can define constraints using *include*, *include-strict*, *exclude*, and *exclude-all* arguments, where each statement can contain up to 10 colors, and define include constraints in both loose and strict sense.

Note You can configure affinity constraints using attribute flags or the Flexible Name Based Tunnel Constraints scheme; however, when configurations for both schemes exist, only the configuration pertaining to the new scheme is applied.

Related Topics

Assigning Color Names to Numeric Values, on page 126
Associating Affinity-Names with TE Links, on page 127
Associating Affinity Constraints for TE Tunnels, on page 128
Configure Flexible Name-based Tunnel Constraints: Example, on page 153

MPLS Traffic Engineering Interarea Tunneling

These topics describe the following new extensions of MPLS-TE:

- Interarea Support, on page 96
- Multiarea Support, on page 96
- Loose Hop Expansion, on page 97
- Loose Hop Reoptimization, on page 97

• Fast Reroute Node Protection, on page 98

Interarea Support

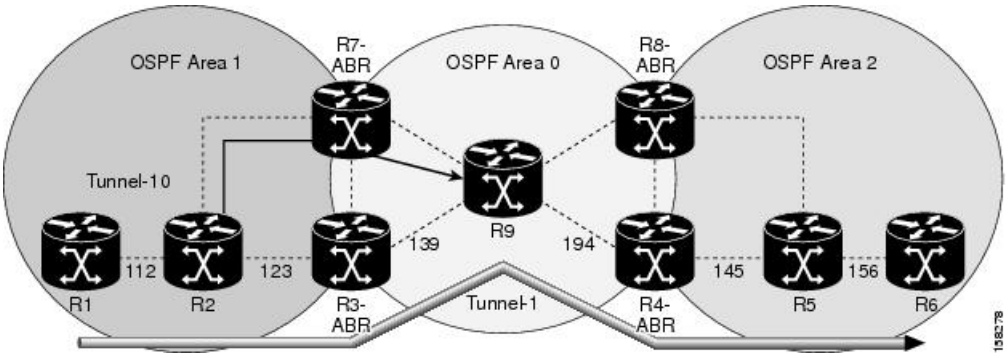
The MPLS-TE interarea tunneling feature allows you to establish P2P tunnels spanning multiple Interior Gateway Protocol (IGP) areas and levels, thereby eliminating the requirement that headend and tailend routers reside in a single area.

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.

Multiarea and Interarea TE are required by the customers running multiple IGP area backbones (primarily for scalability reasons). 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.

Figure 9: Interarea (OSPF) TE Network Diagram

This figure shows a typical interarea TE network.



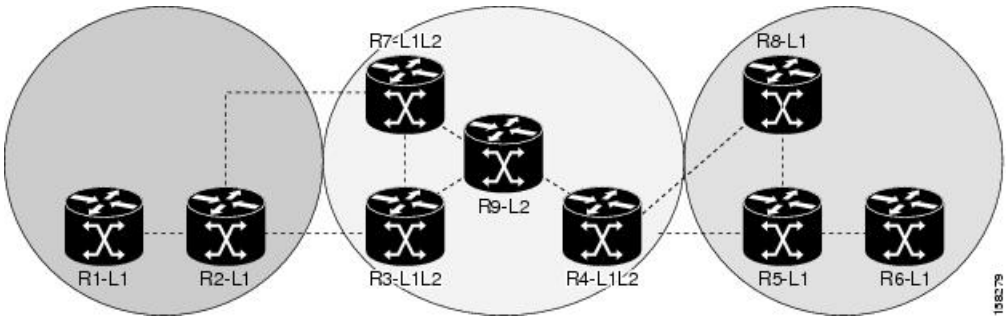
Multiarea Support

Multiarea support allows an area border router (ABR) LSR to support MPLS-TE in more than one IGP area. A TE LSP is still confined to a single area.

Multiarea and Interarea TE are required when you run multiple IGP area backbones. The Multiarea and Interarea TE allows you to:

- Limit the volume of flooded information.
- Reduce the SPF duration.
- Decrease the impact of a link or node failure within an area.

Figure 10: Interlevel (IS-IS) TE Network



As shown in the figure, 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).

Note You can configure multiple areas within an IS-IS Level 1. This is transparent to TE. TE has topology information about the IS-IS level, but not the area ID.

Loose Hop Expansion

Loose hop optimization allows the reoptimization 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. It is the then 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). 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.

Loose Hop Reoptimization

Loose hop reoptimization allows the reoptimization of the tunnels spanning multiple areas and solves the problem which occurs when an MPLS-TE headend does not have visibility into other IGP areas.

Whenever the headend attempts to reoptimize a tunnel, it tries to find a better path to the ABR in the headend area. If a better path is found then the headend initiates the setup of a new LSP. In case a suitable path is not found in the headend area, the headend initiates a querying message. The purpose of this message is to query the ABRs in the areas other than the headend area to check if there exist any better paths in those areas. The purpose of this message is to query the ABRs in the areas other than the headend area, to check if a better

path exists. If a better path does not exist, ABR forwards the query to the next router downstream. Alternatively, if better path is found, ABR responds with a special Path Error to the headend to indicate the existence of a better path outside the headend area. Upon receiving the Path Error that indicates the existence of a better path, the headend router initiates the reoptimization.

ABR Node Protection

Because one IGP area does not have visibility into another IGP area, it is not possible to assign backup to protect ABR node. To overcome this problem, node ID sub-object is added into the record route object of the primary tunnel so that at a PLR node, backup destination address can be checked against primary tunnel record-route object and assign a backup tunnel.

Fast Reroute Node Protection

If a link failure occurs within an area, the upstream router directly connected to the failed link generates an RSVP path error message to the headend. As a response to the message, the headend sends an RSVP path tear message and the corresponding path option is marked as invalid for a specified period and the next path-option (if any) is evaluated.

To retry the ABR immediately, a second path option (identical to the first one) should be configured. Alternatively, the retry period (path-option hold-down, 2 minutes by default) can be tuned to achieve a faster retry.

Related Topics

Protecting MPLS Tunnels with Fast Reroute, on page 114

MPLS-TE Forwarding Adjacency

The MPLS-TE Forwarding Adjacency feature allows a network administrator to handle a traffic engineering, label-switched path (LSP) tunnel as a link in an Interior Gateway Protocol (IGP) network based on the Shortest Path First (SPF) algorithm. A forwarding adjacency can be created between routers regardless of their location in the network.

MPLS-TE Forwarding Adjacency Benefits

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 head end of any TE tunnels.

Related Topics

Configuring MPLS-TE Forwarding Adjacency, on page 131
Configure Forwarding Adjacency: Example, on page 155

MPLS-TE Forwarding Adjacency Restrictions

The MPLS-TE Forwarding Adjacency feature has these restrictions:

- Using the MPLS-TE Forwarding Adjacency increases the size of the IGP database by advertising a TE tunnel as a link.
- The MPLS-TE Forwarding Adjacency is supported by Intermediate System-to-Intermediate System (IS-IS).
- When the MPLS-TE Forwarding Adjacency is enabled on a TE tunnel, the link is advertised in the IGP network as a Type-Length-Value (TLV) 22 without any TE sub-TLV.

- MPLS-TE forwarding adjacency tunnels must be configured bidirectionally.
- Multicast intact is not supported with MPLS-TE Forwarding Adjacency.

MPLS-TE Forwarding Adjacency Prerequisites

Your network must support the following features before enabling the MPLS -TE Forwarding Adjacency feature:

- MPLS
- IP Cisco Express Forwarding
- Intermediate System-to-Intermediate System (IS-IS)

Path Computation Element

Path Computation Element (PCE) solves the specific issue of inter-domain path computation for MPLS-TE label switched path (LSPs), when the head-end router does not possess full network topology information (for example, when the head-end and tail-end routers of an LSP reside in different IGP areas).

PCE uses area border routers (ABRs) to compute a TE LSP spanning multiple IGP areas as well as computation of Inter-AS TE LSP.

PCE is usually used to define an overall architecture, which is made of several components, as follows:

Path Computation Element (PCE)

Represents a software module (which can be a component or application) that enables the router to compute paths applying a set of constraints between any pair of nodes within the router's TE topology database. PCEs are discovered through IGP.

Path Computation Client (PCC)

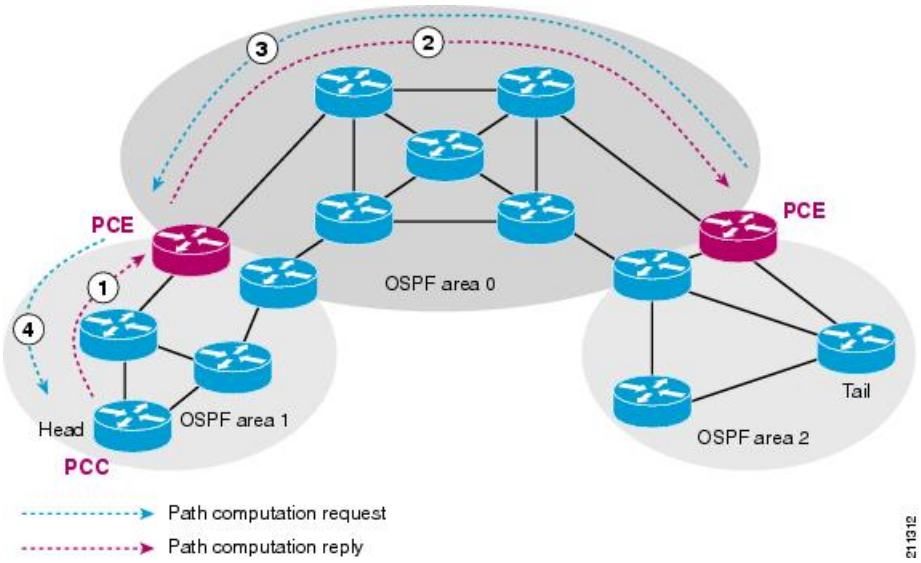
Represents a software module running on a router that is capable of sending and receiving path computation requests and responses to and from PCEs. The PCC is typically an LSR (Label Switching Router).

PCC-PCE communication protocol (PCEP)

Specifies that PCEP is a TCP-based protocol defined by the IETF PCE WG, and defines a set of messages and objects used to manage PCEP sessions and to request and send paths for multi-domain TE LSPs. PCEP is used for communication between PCC and PCE (as well as between two PCEs) and employs IGP extensions to dynamically discover PCE.

Figure 11: Path Computation Element Network Diagram

This figure shows a typical PCE implementation.



Path computation elements provides support for the following message types and objects:

- Message types: Open, PCReq, PCRep, PCErr, Close
- Objects: OPEN, CLOSE, RP, END-POINT, LSPA, BANDWIDTH, METRIC, and NO-PATH

Related Topics

Configuring a Path Computation Client, on page 132
Configuring a Path Computation Element Address, on page 133
Configuring PCE Parameters, on page 134
Configure PCE: Example, on page 156

Policy-Based Tunnel Selection

These topics provide information about policy-based tunnel selection (PBTS):

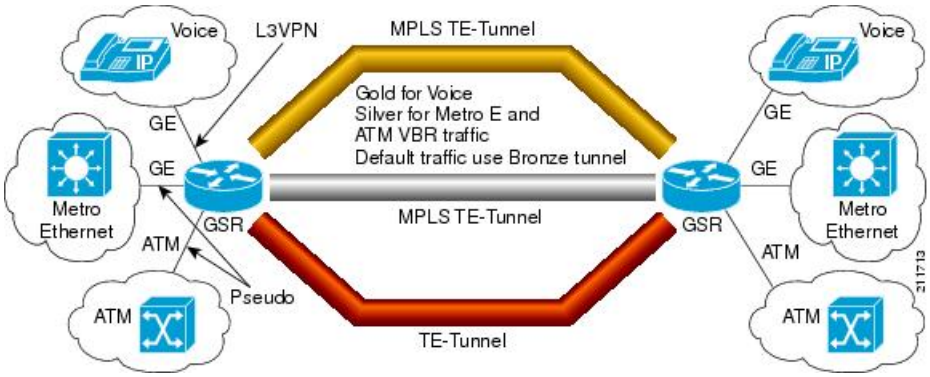
Policy-Based Tunnel Selection

Policy-Based Tunnel Selection (PBTS) provides a mechanism that lets you direct traffic into specific TE tunnels based on different criteria. PBTS will benefit Internet service providers (ISPs) who carry voice and data traffic through their MPLS and MPLS/VPN networks, who want to route this traffic to provide optimized voice service.

PBTS works by selecting tunnels based on the classification criteria of the incoming packets, which are based on the IP precedence, experimental (EXP), or type of service (ToS) field in the packet.

Figure 12: Policy-Based Tunnel Selection Implementation

This figure illustrates a PBTS implementation.



PBTS is supported on the ingress interface and any of the L3 interfaces (physical, sub-interface, and bundle interface).

PBTS supports modification of the class-map and forward-group to TE association.

Related Topics

Configuring Policy-based Tunnel Selection, on page 136

Policy-Based Tunnel Selection Functions

The following PBTS functions are supported:

- IPv4 traffic arrives unlabeled on the VRF interface and the non-VRF interface.
- MPLS traffic is supported on the VRF interface and the non-VRF interface.
- Load balancing across multiple TE tunnels with the same traffic class attribute is supported.
- Selected TE tunnels are used to service the lowest tunnel class as default tunnels.
- LDP over TE tunnel and single-hop TE tunnel are supported.
- Both Interior Gateway Protocol (IGP) and Label Distribution Protocol (LDP) paths are used as the default path for all traffic that belongs to a class that is not configured on the TE tunnels.
- According to the quality-of-service (QoS) policy, tunnel selection is based on the outgoing experimental (EXP) value and the remarked EXP value.

Related Topics

Configuring Policy-based Tunnel Selection, on page 136

PBTS Restrictions

When implementing PBTS, the following restrictions are listed:

- When QoS EXP remarking on an interface is enabled, the EXP value is used to determine the egress tunnel interface, not the incoming EXP value.
- Egress-side remarking does not affect PBTS tunnel selection.
- When no default tunnel is available for forwarding, traffic is dropped.

MPLS-TE Automatic Bandwidth

The MPLS-TE automatic bandwidth feature measures the traffic in a tunnel and periodically adjusts the signaled bandwidth for the tunnel.

These topics provide information about MPLS-TE automatic bandwidth:

MPLS-TE Automatic Bandwidth Overview

MPLS-TE automatic bandwidth is configured on individual Label Switched Paths (LSPs) at every head-end. MPLS-TE monitors the traffic rate on a tunnel interface. Periodically, MPLS-TE 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 upon either the largest average output rate that was noticed during a certain interval, or a configured maximum bandwidth value.

This table lists the automatic bandwidth functions.

Table 6: Automatic Bandwidth Variables

Function	Command	Description	Default Value
Application frequency	**application** command	Configures how often the tunnel bandwidths changed for each tunnel. The application period is the period of A minutes between the bandwidth applications during which the output rate collection is done.	24 hours
Requested bandwidth	**bw-limit** command	Limits the range of bandwidth within the automatic-bandwidth feature that can request a bandwidth.	0 Kbps
Collection frequency	**auto-bw collect** command	Configures how often the tunnel output rate is polled globally for all tunnels.	5 min
Highest collected bandwidth	—	You cannot configure this value.	—
Delta	—	You cannot configure this value.	—

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 reoptimized. Then, the bandwidth samples are cleared to record the new largest output rate at the next interval.

When reoptimizing the LSP with the new bandwidth, a new path request is generated. If the new bandwidth is not available, the last good LSP continues to be 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.

Note When more than 100 tunnels are **auto-bw** enabled, the algorithm will jitter the first application of every tunnel by a maximum of 20% (max 1hour). The algorithm does this to avoid too many tunnels running auto bandwidth applications at the same time.

If a tunnel is shut down, and is later brought again, the adjusted bandwidth is lost and the tunnel is brought back with the initial configured bandwidth. In addition, the application period is reset when the tunnel is brought back.

Related Topics

Configuring the Collection Frequency, on page 138
Configuring the Automatic Bandwidth Functions, on page 139
Configure Automatic Bandwidth: Example, on page 157

Adjustment Threshold

Adjustment Threshold is defined as a percentage of the current tunnel bandwidth and an absolute (minimum) bandwidth. Both thresholds must be fulfilled for the automatic bandwidth to resignal the tunnel. The tunnel bandwidth is resized only if the difference between the largest sample output rate and the current tunnel bandwidth is larger than the adjustment thresholds.

For example, assume that the automatic bandwidth is enabled on a tunnel in which the highest observed bandwidth B is 30 Mbps. Also, assume that the tunnel was initially configured for 45 Mbps. Therefore, the difference is 15 mbit/s. Now, assuming the default adjustment thresholds of 10% and 10kbps, the tunnel is signalled with 30 Mbps when the application timer expires. This is because 10% of 45Mbit/s is 4.5 Mbit/s, which is smaller than 15 Mbit/s. The absolute threshold, which by default is 10kbps, is also crossed.

Overflow Detection

Overflow detection is used if a bandwidth must be resized as soon as an overflow condition is detected, without having to wait for the expiry of an automatic bandwidth application frequency interval.

For overflow detection one configures a limit N, a percentage threshold Y% and optionally, a minimum bandwidth threshold Z. The percentage threshold is defined as the percentage of the actual signalled tunnel bandwidth. When the difference between the measured bandwidth and the actual bandwidth are both larger than Y% and Z threshold, for N consecutive times, then the system triggers an overflow detection.

The bandwidth adjustment by the overflow detection is triggered only by an increase of traffic volume through the tunnel, and not by a decrease in the traffic volume. When you trigger an overflow detection, the automatic bandwidth application interval is reset.

By default, the overflow detection is disabled and needs to be manually configured.

Underflow Detection

Underflow detection is used when the bandwidth on a tunnel drops significantly, which is similar to overflow but in reverse.

Underflow detection applies the highest bandwidth value from the samples which triggered the underflow. For example, if you have an underflow limit of three, and the following samples trigger the underflow for 10 kbps, 20 kbps, and 15 kbps, then, 20 kbps is applied.

Unlike overflow, the underflow count is not reset across an application period. For example, with an underflow limit of three, you can have the first two samples taken at the end of an application period and then the underflow gets triggered by the first sample of the next application period.

Restrictions for MPLS-TE Automatic Bandwidth

When the automatic bandwidth cannot update the tunnel bandwidth, the following restrictions are listed:

- Tunnel is in a fast reroute (FRR) backup, active, or path protect active state. This occurs because of the assumption that protection is a temporary state, and there is no need to reserve the bandwidth on a backup tunnel. You should prevent taking away the bandwidth from other primary or backup tunnels.
- Reoptimization fails to occur during a lockdown. In this case, the automatic bandwidth does not update the bandwidth unless the bandwidth application is manually triggered by using the **mpls traffic-eng auto-bw apply** command in EXEC mode.

Point-to-Multipoint Traffic-Engineering

Point-to-Multipoint Traffic-Engineering Overview

The Point-to-Multipoint (P2MP) Resource Reservation Protocol-Traffic Engineering (RSVP-TE) solution allows service providers to implement IP multicast applications, such as IPTV and real-time video, broadcast over the MPLS label switch network. The RSVP-TE protocol is extended to signal point-to-point (P2P) and P2MP label switched paths (LSPs) across the MPLS networks.

By using RSVP-TE extensions as defined in RFC 4875, multiple subLSPs are signaled for a given TE source. The P2MP tunnel is considered as a set of Source-to-Leaf (S2L) subLSPs that connect the TE source to multiple leaf Provider Edge (PE) nodes.

At the TE source, the ingress point of the P2MP-TE tunnel, IP multicast traffic is encapsulated with a unique MPLS label, which is associated with the P2MP-TE tunnel. The traffic continues to be label-switched in the P2MP tree. If needed, the labeled packet is replicated at branch nodes along the P2MP tree. When the labeled packet reaches the egress leaf (PE) node, the MPLS label is removed and forwarded onto the IP multicast tree across the PE-CE link.

To enable end-to-end IP multicast connectivity, RSVP is used in the MPLS-core for P2MP-TE signaling and PIM is used for PE-CE link signaling.

- All edge routers are running PIM-SSM or Source-Specific Multicast (SSM) to exchange multicast routing information with the directly-connected Customer Edge (CE) routers.
- In the MPLS network, RSVP P2MP-TE replaces PIM as the tree building mechanism, RSVP-TE grafts or prunes a given P2MP tree when the end-points are added or removed in the TE source configuration (explicit user operation).

These are the definitions for Point-to-Multipoint (P2MP) tunnels:

Source

Configures the node in which Label Switched Path (LSP) signaling is initiated.

Mid-point

Specifies the transit node in which LSP signaling is processed (for example, not a source or receiver).

Receiver, Leaf, and Destination

Specifies the node in which LSP signaling ends.

Branch Point

Specifies the node in which packet replication is performed.

Source-to-Leaf (S2L) SubLSP

Specifies the P2MP-TE LSP segment that runs from the source to one leaf.

Note Cisco NCS 6000 Series Routers supports only P2MP TE mid-point functionality. The MPLS and the multicast packages are required the mid point router for the P2MP TE feature to work.

Point-to-Multipoint Traffic-Engineering Features

- P2MP RSVP-TE (RFC 4875) is supported. RFC 4875 is based on nonaggregate signaling; for example, per S2L signaling. Only P2MP LSP is supported.
- **interface tunnel-mte** command identifies the P2MP interface type on the Head-end.
- P2MP tunnel setup is supported with label replication.
- Fast-Reroute (FRR) protection is supported with sub-50 msec for traffic loss.
- Explicit routing is supported by using under utilized links.
- Reoptimization is supported by calculating a better set of paths to the destination with no traffic loss.

  Note Per-S2L reoptimization is not supported.

- IPv4 and IPv6 payloads are supported.
- IPv4 and IPv6 multicast forwarding are supported on a P2MP tunnel interface through a static IGMP and MLD group configuration on the Head-end.
- Both IP multicast and P2MP Label Switch Multicast (LSM) coexist in the same network; therefore, both use the same forwarding plane (LFIB or MPLS Forwarding Infrastructure [MFI]).
- P2MP label replication supports only Source-Specific Multicast (SSM) traffic. SSM configuration supports the default value, none.
- Static mapping for multicast groups to the P2MP-TE tunnel is required on the Head-end.

Point-to-Multipoint Traffic-Engineering Benefits

- Single point of traffic control ensures that signaling and path engineering parameters (for example, protection and diversity) are configured only at the TE source node.
- Ability to configure explicit paths to enable optimized traffic distribution and prevention of single point of failures in the network.
- Link protection of MPLS-labeled traffic traversing branch paths of the P2MP-TE tree.
- Ability to do bandwidth Admission Control (AC) during set up and signaling of P2MP-TE paths in the MPLS network.

Related Topics

Point-to-Multipoint RSVP-TE , on page 106

Point-to-Multipoint RSVP-TE

RSVP-TE signals a P2MP tunnel base that is based on a manual configuration. If all Source-to-Leaf (S2L)s use an explicit path, the P2MP tunnel creates a static tree that follows a predefined path based on a constraint such as a deterministic Label Switched Path (LSP). If the S2L uses a dynamic path, RSVP-TE creates a P2MP tunnel base on the best path in the RSVP-TE topology. RSVP-TE supports bandwidth reservation for constraint-based routing.

When an explicit path option is used, specify both the local and peer IP addresses in the explicit path option, provided the link is a GigabitEthernet or a TenGigE based interface. For point-to-point links like POS or bundle POS, it is sufficient to mention the remote or peer IP address in the explicit path option.

RSVP-TE distributes stream information in which the topology tree does not change often (where the source and receivers are). For example, large scale video distribution between major sites is suitable for a subset of multicast applications. Because multicast traffic is already in the tunnel, the RSVP-TE tree is protected as long as you build a backup path.

Fast-Reroute (FRR) capability is supported for P2MP RSVP-TE by using the unicast link protection. You can choose the type of traffic to go to the backup link.

The P2MP tunnel is applicable for all TE Tunnel destination (IntraArea and InterArea). Inter-AS is not supported.

The P2MP tunnel is signaled by the dynamic and explicit path option in the IGP intra area. Only interArea and interAS, which are used for the P2MP tunnels, are signaled by the verbatim path option.

Related Topics

Point-to-Multipoint Fast Reroute, on page 106

Point-to-Multipoint Fast Reroute

MPLS-TE Fast Reroute (FRR) is a mechanism to minimize interruption in traffic delivery to a TE Label Switched Path (LSP) destination as a result of link failures. FRR enables temporarily fast switching of LSP traffic along an alternative backup path around a network failure, until the TE tunnel source signals a new end-to-end LSP.

Both Point-to-Point (P2P) and P2MP-TE support only the Facility FRR method from RFC 4090.

P2P LSPs are used to backup P2MP S2L (source 2 Leaf). Only link and bandwidth protection for P2MP S2Ls are supported. Node protection is not supported.

MPLS-TE link protection relies on the fact that labels for all primary LSPs and subLSPs are using the MPLS global label allocation. For example, one single (global) label space is used for all MPLS-TE enabled physical interfaces on a given MPLS LSP.

Related Topics

Point-to-Multipoint Traffic-Engineering Overview, on page 104
Point-to-Multipoint RSVP-TE , on page 106

Point-to-Multipoint Label Switch Path

The Point-to-Multipoint Label Switch Path (P2MP LSP) has only a single root, which is the Ingress Label Switch Router (LSR). The P2MP LSP is created based on a receiver that is connected to the Egress LSR. The Egress LSR initiates the creation of the tree (for example, tunnel grafting or pruning is done by performing an individual sub-LSP operation) by creating the Forwarding Equivalency Class (FEC) and Opaque Value.

Note Grafting and pruning operate on a per destination basis.

The Opaque Value contains the stream information that uniquely identifies the tree to the root. To receive label switched multicast packets, the Egress Provider Edge (PE) indicates to the upstream router (the next hop closest to the root) which label it uses for the multicast source by applying the label mapping message.

The upstream router does not need to have any knowledge of the source; it needs only the received FEC to identify the correct P2MP LSP. If the upstream router does not have any FEC state, it creates it and installs the assigned downstream outgoing label into the label forwarding table. If the upstream router is not the root of the tree, it must forward the label mapping message to the next hop upstream. This process is repeated hop-by-hop until the root is reached.

By using downstream allocation, the router that wants to receive the multicast traffic assigns the label for it. The label request, which is sent to the upstream router, is similar to an unsolicited label mapping (that is, the upstream does not request it). The upstream router that receives that label mapping uses the specific label to send multicast packets downstream to the receiver. The advantage is that the router, which allocates the labels, does not get into a situation where it has the same label for two different multicast sources. This is because it manages its own label space allocation locally.

Interarea P2MP Path Expansion within a Domain

Interarea P2MP (Point-to-Multipoint) path expansion within a domain feature matches the domain of the subsequent auto-discovered ABR (Area Border Router) with the domain of the incoming interface where the Path message is received. This feature restricts the ERO (Explicit Route Object) expansion using the same domain as associated with the incoming interface where the Path message is received. This restriction applies to both loose-hop ABR and dynamically discovered ABR.

Configure this feature using the **path-selection loose-expansion domain-match** command in MPLS-TE configuration.

Interarea P2MP path expansion within a domain configuration applies to:

- All interarea TE (Traffic Engineering) path expansions on the ABR node
- Both P2P (Point-to-Point) and P2MP interarea TE LSPs
- Midpoint nodes

Limitation

The ERO expansion domain-match is not supported for multiple incoming IGPs.

How to Implement Traffic Engineering

Traffic engineering requires coordination among several global neighbor routers, creating traffic engineering tunnels, setting up forwarding across traffic engineering tunnels, setting up FRR, and creating differential service.

These procedures are used to implement MPLS-TE:

Building MPLS-TE Topology

Perform this task to configure MPLS-TE topology (required for traffic engineering tunnel operations).

Before you begin

Before you start to build the MPLS-TE topology, you must have enabled:

- IGP such as OSPF or IS-IS for MPLS-TE.
- MPLS Label Distribution Protocol (LDP).
- RSVP on the port interface.
- 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.
- If you are going to use nondefault holdtime or intervals, you must decide the values to which they are set.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **interface** *type interface-path-id*
4. **exit**
5. **exit**
6. **router ospf** *process-name*
7. **area** *area-id*
8. **exit**
9. **mpls traffic-eng router-id** *ip-address*
10. **commit**
11. (Optional) **show mpls traffic-eng topology**
12. (Optional) **show mpls traffic-eng link-management advertisements**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 3	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te)#interface POS0/6/0/0`
	`RP/0/RP0/CPU0:router(config-mpls-te-if)#`
		Enables traffic engineering on a particular interface on the originating node and enters MPLS-TE interface configuration mode.

Step 4	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te-if)# exit`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Exits the current configuration mode.

Step 5	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te)# exit`
	`RP/0/RP0/CPU0:router(config)#`
		Exits the current configuration mode.

Step 6	**router ospf** *process-name* **Example:** `RP/0/RP0/CPU0:router(config)# router ospf 1`	Enters a name for the OSPF process.
Step 7	**area** *area-id* **Example:** `RP/0/RP0/CPU0:router(config-router)# area 0`	Configures an area for the OSPF process. • Backbone areas have an area ID of 0. • Non-backbone areas have a non-zero area ID.

Step 8	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-ospf-ar)# exit`
	`RP/0/RP0/CPU0:router(config-ospf)#`
		Exits the current configuration mode.

	Command or Action	Purpose
Step 9	**mpls traffic-eng router-id** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-ospf)# mpls traffic-eng router-id 192.168.70.1`	Sets the MPLS-TE loopback interface.
Step 10	**commit**	
Step 11	(Optional) **show mpls traffic-eng topology** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng topology`	Verifies the traffic engineering topology.
Step 12	(Optional) **show mpls traffic-eng link-management advertisements** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng link-management advertisements`	Displays all the link-management advertisements for the links on this node.

Related Topics

How MPLS-TE Works, on page 89
Build MPLS-TE Topology and Tunnels: Example, on page 151

Creating an MPLS-TE Tunnel

Creating an MPLS-TE tunnel is a process of customizing the traffic engineering to fit your network topology.

Perform this task to create an MPLS-TE tunnel after you have built the traffic engineering topology.

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.
- If you are going to use nondefault holdtime or intervals, you must decide the values to which they are set.

SUMMARY STEPS

1. **configure**

2. **interface tunnel-te** *tunnel-id*
3. **destination** *ip-address*
4. **ipv4 unnumbered** *type interface-path-id*
5. **path-option** *preference - priority* **dynamic**
6. **signalled- bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*}
7. **commit**
8. (Optional) **show mpls traffic-eng tunnels**
9. (Optional) **show ipv4 interface brief**
10. (Optional) **show mpls traffic-eng link-management admission-control**

DETAILED STEPS

	Command or Action	Purpose
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	**destination** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125`	Assigns a destination address on the new tunnel. The destination address is the remote node's MPLS-TE router ID.
Step 4	**ipv4 unnumbered** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0`	Assigns a source address so that forwarding can be performed on the new tunnel. Loopback is commonly used as the interface type.
Step 5	**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 6	**signalled- bandwidth** {*bandwidth* [**class-type** *ct*]	**sub-pool** *bandwidth*} **Example:** `RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 100`	Sets the CT0 bandwidth required on this interface. Because the default tunnel priority is 7, tunnels use the default TE class map (namely, class-type 1, priority 7).

	Command or Action	Purpose
Step 7	**commit**	
Step 8	(Optional) **show mpls traffic-eng tunnels** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels`	Verifies that the tunnel is connected (in the UP state) and displays all configured TE tunnels.
Step 9	(Optional) **show ipv4 interface brief** **Example:** `RP/0/RP0/CPU0:router# show ipv4 interface brief`	Displays all TE tunnel interfaces.
Step 10	(Optional) **show mpls traffic-eng link-management admission-control** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng link-management admission-control`	Displays all the tunnels on this node.

Related Topics

How MPLS-TE Works, on page 89
Build MPLS-TE Topology and Tunnels: Example, on page 151
Building MPLS-TE Topology, on page 108

Configuring Forwarding over the MPLS-TE Tunnel

Perform this task to configure forwarding over the MPLS-TE tunnel created in the previous task . This task allows MPLS packets to be forwarded on the link between network neighbors.

Before you begin

The following prerequisites are required to configure forwarding over the 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.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **ipv4 unnumbered** *type interface-path-id*
4. **autoroute announce**

5. **exit**
6. **router static address-family ipv4 unicast** *prefix mask ip-address interface type*
7. **commit**
8. (Optional) **ping** {*ip-address* | *hostname*}
9. (Optional) **show mpls traffic-eng autoroute**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface tunnel-te 1`	Enters MPLS-TE interface configuration mode.
Step 3	**ipv4 unnumbered** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0`	Assigns a source address so that forwarding can be performed on the new tunnel.
Step 4	**autoroute announce** **Example:** `RP/0/RP0/CPU0:router(config-if)# autoroute announce`	Enables messages that notify the neighbor nodes about the routes that are forwarding.
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits the current configuration mode.
Step 6	**router static address-family ipv4 unicast** *prefix mask ip-address interface type* **Example:** `RP/0/RP0/CPU0:router(config)# router static address-family ipv4 unicast 2.2.2.2/32 tunnel-te 1`	Enables a route using IP version 4 addressing, identifies the destination address and the tunnel where forwarding is enabled. This configuration is used for static routes when the **autoroute announce** command is not used.
Step 7	**commit**	
Step 8	(Optional) **ping** {*ip-address* | *hostname*} **Example:**	Checks for connectivity to a particular IP address or host name.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# ping 192.168.12.52`	
Step 9	(Optional) **show mpls traffic-eng autoroute** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng autoroute`	Verifies forwarding by displaying what is advertised to IGP for the TE tunnel.

Related Topics

Overview of MPLS Traffic Engineering, on page 88
Creating an MPLS-TE Tunnel, on page 110

Protecting MPLS Tunnels with Fast Reroute

Perform this task to protect MPLS-TE tunnels, as created in the previous task.

Note Although this task is similar to the previous task, its importance makes it necessary to present as part of the tasks required for traffic engineering on Cisco IOS XR software.

Before you begin

The following prerequisites are required to protect MPLS-TE tunnels:

- 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.
- You must first configure a primary tunnel.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **fast-reroute**
4. **exit**
5. **mpls traffic-eng**
6. **interface** *type interface-path-id*
7. **backup-path tunnel-te** *tunnel-number*
8. **exit**
9. **exit**
10. **interface tunnel-te** *tunnel-id*

11. **backup-bw** {*backup bandwidth* | **sub-pool** {*bandwidth* | **unlimited**} | **global-pool** {*bandwidth* | **unlimited**} }
12. **ipv4 unnumbered** *type interface-path-id*
13. **path-option** *preference-priority* {**explicit name** *explicit-path-name*}
14. **destination** *ip-address*
15. **commit**
16. (Optional) **show mpls traffic-eng tunnels backup**
17. (Optional) **show mpls traffic-eng tunnels protection frr**
18. (Optional) **show mpls traffic-eng fast-reroute database**

DETAILED STEPS

	Command or Action	Purpose
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	**fast-reroute** **Example:** `RP/0/RP0/CPU0:router(config-if)# fast-reroute`	Enables fast reroute.
Step 4	**exit** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits the current configuration mode.

Step 5	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 6	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te)# interface pos0/6/0/0`
	`RP/0/RP0/CPU0:router(config-mpls-te-if)#`
		Enables traffic engineering on a particular interface on the originating node.

Step 7	**backup-path tunnel-te** *tunnel-number* **Example:**	Sets the backup path to the backup tunnel.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-mpls-te-if)# backup-path tunnel-te 2`	

Step 8	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te-if)# exit`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Exits the current configuration mode.

Step 9	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te)# exit`
	`RP/0/RP0/CPU0:router(config)#`
		Exits the current configuration mode.

Step 10	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface tunnel-te 2`	Configures an MPLS-TE tunnel interface.
Step 11	**backup-bw** {*backup bandwidth* | **sub-pool** {*bandwidth* | **unlimited**} | **global-pool** {*bandwidth* | **unlimited**} } **Example:** `RP/0/RP0/CPU0:router(config-if)#backup-bw global-pool 5000`	Sets the CT0 bandwidth required on this interface. **Note** Because the default tunnel priority is 7, tunnels use the default TE class map.
Step 12	**ipv4 unnumbered** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0`	Assigns a source address to set up forwarding on the new tunnel.
Step 13	**path-option** *preference-priority* {**explicit name** *explicit-path-name*} **Example:** `RP/0/RP0/CPU0:router(config-if)# path-option 1 explicit name backup-path`	Sets the path option to explicit with a given name (previously configured) and assigns the path ID.
Step 14	**destination** *ip-address* **Example:**	Assigns a destination address on the new tunnel.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125`	• Destination address is the remote node's MPLS-TE router ID. • Destination address is the merge point between backup and protected tunnels. **Note** When you configure TE tunnel with multiple protection on its path and merge point is the same node for more than one protection, you must configure record-route for that tunnel.
Step 15	**commit**	
Step 16	(Optional) **show mpls traffic-eng tunnels backup** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels backup`	Displays the backup tunnel information.
Step 17	(Optional) **show mpls traffic-eng tunnels protection frr** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels protection frr`	Displays the tunnel protection information for Fast-Reroute (FRR).
Step 18	(Optional) **show mpls traffic-eng fast-reroute database** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng fast-reroute database`	Displays the protected tunnel state (for example, the tunnel's current ready or active state).

Related Topics

Fast Reroute, on page 93
Fast Reroute Node Protection, on page 98
Creating an MPLS-TE Tunnel, on page 110
Configuring Forwarding over the MPLS-TE Tunnel , on page 112

Configuring a Prestandard DS-TE Tunnel

Perform this task to configure a Prestandard DS-TE tunnel.

Before you begin

The following prerequisites are required to configure a Prestandard 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.

SUMMARY STEPS

1. **configure**
2. **rsvp interface** *type interface-path-id*
3. **bandwidth** [*total reservable bandwidth*] [**bc0** *bandwidth*] [**global-pool** *bandwidth*] [**sub-pool** *reservable-bw*]
4. **exit**
5. **exit**
6. **interface tunnel-te** *tunnel-id*
7. **signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*}
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# rsvp interface pos0/6/0/0`	Enters RSVP configuration mode and selects an RSVP interface.
Step 3	**bandwidth** [*total reservable bandwidth*] [**bc0** *bandwidth*] [**global-pool** *bandwidth*] [**sub-pool** *reservable-bw*] **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth 100 150 sub-pool 50`	Sets the reserved RSVP bandwidth available on this interface by using the prestandard DS-TE mode. The range for the *total reserve bandwidth* argument is 0 to 4294967295. Physical interface bandwidth is not used by MPLS-TE.

Step 4	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-if)# exit`
	`RP/0/RP0/CPU0:router(config-rsvp)#`
		Exits the current configuration mode.

Step 5	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp)# exit`
	`RP/0/RP0/CPU0:router(config)#`
		Exits the current configuration mode.

	Command or Action	Purpose
Step 6	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface tunnel-te 2`	Configures an MPLS-TE tunnel interface.
Step 7	**signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*} **Example:** `RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth sub-pool 10`	Sets the bandwidth required on this interface. Because the default tunnel priority is 7, tunnels use the default TE class map (namely, class-type 1, priority 7).
Step 8	**commit**	

Related Topics

Configuring Traffic Engineering Tunnel Bandwidth, on page 57
Prestandard DS-TE Mode, on page 90
Configure IETF DS-TE Tunnels: Example, on page 152

Configuring an IETF DS-TE Tunnel Using RDM

Perform this task to create an IETF mode DS-TE tunnel using RDM.

Before you begin

The following prerequisites are required to create an 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.

SUMMARY STEPS

1. **configure**
2. **rsvp interface** *type interface-path-id*
3. **bandwidth rdm** {*total-reservable-bw* | **bc0** | **global-pool**} {**sub-pool** | **bc1** *reservable-bw*}
4. **exit**
5. **exit**
6. **mpls traffic-eng**
7. **ds-te mode ietf**
8. **exit**
9. **interface tunnel-te** *tunnel-id*
10. **signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*}

11. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# rsvp interface pos0/6/0/0`	Enters RSVP configuration mode and selects an RSVP interface.
Step 3	**bandwidth rdm** {*total-reservable-bw* | **bc0** | **global-pool**} {**sub-pool** | **bc1** *reservable-bw*} **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth rdm 100 150`	Sets the reserved RSVP bandwidth available on this interface by using the Russian Doll Model (RDM) bandwidth constraints model. The range for the *total reserve bandwidth* argument is 0 to 4294967295. **Note** Physical interface bandwidth is not used by MPLS-TE.

Step 4	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-if)# exit`
	`RP/0/RP0/CPU0:router(config-rsvp)`
		Exits the current configuration mode.

Step 5	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp) exit`
	`RP/0/RP0/CPU0:router(config)`
		Exits the current configuration mode.

Step 6	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 7	**ds-te mode ietf** **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# ds-te mode ietf`	Enables IETF DS-TE mode and default TE class map. IETF DS-TE mode is configured on all network nodes.

	Command or Action	Purpose
Step 8	**exit** **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# exit`	Exits the current configuration mode.

Step 9	**interface tunnel-te** *tunnel-id*
	Example:
	`RP/0/RP0/CPU0:router(config)# interface tunnel-te 4`
	`RP/0/RP0/CPU0:router(config-if)#`
		Configures an MPLS-TE tunnel interface.

Step 10	**signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*} **Example:** `RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 10 class-type 1`	Configures the bandwidth required for an MPLS TE tunnel. Because the default tunnel priority is 7, tunnels use the default TE class map (namely, class-type 1, priority 7).
Step 11	**commit**	

Related Topics

Configuring Traffic Engineering Tunnel Bandwidth, on page 57
Russian Doll Bandwidth Constraint Model, on page 91

Configuring an IETF DS-TE Tunnel Using MAM

Perform this task to configure an IETF mode differentiated services traffic engineering tunnel using the Maximum Allocation Model (MAM) bandwidth constraint model.

Before you begin

The following prerequisites are required to configure an IETF mode differentiated services traffic engineering tunnel using the MAM bandwidth constraint model:

- 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.

SUMMARY STEPS

1. **configure**
2. **rsvp interface** *type interface-path-id*

3. **bandwidth mam** {*total reservable bandwidth* | **max-reservable-bw** *maximum-reservable-bw*} [**bc0** *reservable bandwidth*] [**bc1** *reservable bandwidth*]
4. **exit**
5. **exit**
6. **mpls traffic-eng**
7. **ds-te mode ietf**
8. **ds-te bc-model mam**
9. **exit**
10. **interface tunnel-te** *tunnel-id*
11. **signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*}
12. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**rsvp interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# rsvp interface pos0/6/0/0`	Enters RSVP configuration mode and selects the RSVP interface.
Step 3	**bandwidth mam** {*total reservable bandwidth*	**max-reservable-bw** *maximum-reservable-bw*} [**bc0** *reservable bandwidth*] [**bc1** *reservable bandwidth*] **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth mam max-reservable-bw 400 bc0 300 bc1 200`	Sets the reserved RSVP bandwidth available on this interface. **Note** Physical interface bandwidth is not used by MPLS-TE.

Step 4	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp-if)# exit`
	`RP/0/RP0/CPU0:router(config-rsvp)#`
		Exits the current configuration mode.

Step 5	**exit**
	Example:
	`RP/0/RP0/CPU0:router(config-rsvp)# exit`
	`RP/0/RP0/CPU0:router(config)#`
		Exits the current configuration mode.

Step 6	**mpls traffic-eng** **Example:**	Enters MPLS-TE configuration mode.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng` `RP/0/RP0/CPU0:router(config-mpls-te)#`	
Step 7	**ds-te mode ietf** **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# ds-te mode ietf`	Enables IETF DS-TE mode and default TE class map. Configure IETF DS-TE mode on all nodes in the network.
Step 8	**ds-te bc-model mam** **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# ds-te bc-model mam`	Enables the MAM bandwidth constraint model globally.
Step 9	**exit** **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# exit`	Exits the current configuration mode.

Step 10	**interface tunnel-te** *tunnel-id*
	Example:
	`RP/0/RP0/CPU0:router(config)# interface tunnel-te 4`
	`RP/0/RP0/CPU0:router(config-if)#`
		Configures an MPLS-TE tunnel interface.

Step 11	**signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*} **Example:** `RP/0/RP0/CPU0:router(config-rsvp-if)# signalled-bandwidth 10 class-type 1`	Configures the bandwidth required for an MPLS TE tunnel. Because the default tunnel priority is 7, tunnels use the default TE class map (namely, class-type 1, priority 7).
Step 12	**commit**	

Related Topics

Configuring Traffic Engineering Tunnel Bandwidth, on page 57
Maximum Allocation Bandwidth Constraint Model, on page 91

Configuring MPLS -TE and Fast-Reroute on OSPF

Perform this task to configure MPLS-TE and Fast Reroute (FRR) on OSPF.

Before you begin

Note Only point-to-point (P2P) interfaces are supported for OSPF multiple adjacencies. These may be either native P2P interfaces or broadcast interfaces on which the **OSPF P2P configuration** command is applied to force them to behave as P2P interfaces as far as OSPF is concerned. This restriction does not apply to IS-IS.

The tunnel-te interface is not supported under IS-IS.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **path-option** [**protecting**] *preference-priority* {**dynamic** [**pce** [**address ipv4** *address*] | **explicit** {**name** *pathname* | **identifier** *path-number* } } [**isis** *instance name* {**level** *level*}] [**ospf** *instance name* {**area** *area ID*}]] [**verbatim**] [**lockdown**]
4. Repeat Step 3 as many times as needed.
5. **commit**
6. **show mpls traffic-eng tunnels** [*tunnel-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**interface tunnel-te** *tunnel-id*
	Example:
	`RP/0/RP0/CPU0:router(config)# interface tunnel-te 1`
	`RP/0/RP0/CPU0:router(config-if)#`
		Configures an MPLS-TE tunnel interface. The range for the tunnel ID number is 0 to 65535.

Step 3	**path-option** [**protecting**] *preference-priority* {**dynamic** [**pce** [**address ipv4** *address*] | **explicit** {**name** *pathname* | **identifier** *path-number* } } [**isis** *instance name* {**level** *level*}] [**ospf** *instance name* {**area** *area ID*}]] [**verbatim**] [**lockdown**] **Example:** `RP/0/RP0/CPU0:router(config-if)# path-option 1 explicit identifier 6 ospf green area 0`	Configures an explicit path option for an MPLS-TE tunnel. OSPF is limited to a single OSPF instance and area.
Step 4	Repeat Step 3 as many times as needed. **Example:** `RP/0/RP0/CPU0:router(config-if)# path-option 2 explicit name 234 ospf 3 area 7 verbatim`	Configures another explicit path option.

	Command or Action	Purpose
Step 5	**commit**	
Step 6	**show mpls traffic-eng tunnels** [*tunnel-number*] **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels 1`	Displays information about MPLS-TE tunnels.

Configuring the Ignore Integrated IS-IS Overload Bit Setting in MPLS-TE

Perform this task to configure an overload node avoidance in MPLS-TE. When the overload bit is enabled, tunnels are brought down when the overload node is found in the tunnel path.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **path-selection ignore overload**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 3	**path-selection ignore overload**
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
	`path-selection ignore overload`
		Ignores the Intermediate System-to-Intermediate System (IS-IS) overload bit setting for MPLS-TE.

Step 4	**commit**	

Related Topics

Ignore Intermediate System-to-Intermediate System Overload Bit Setting in MPLS-TE, on page 94
Configure the Ignore IS-IS Overload Bit Setting in MPLS-TE: Example, on page 153

Configuring Flexible Name-based Tunnel Constraints

To fully configure MPLS-TE flexible name-based tunnel constraints, you must complete these high-level tasks in order:

1. Assigning Color Names to Numeric Values, on page 126
2. Associating Affinity-Names with TE Links, on page 127
3. Associating Affinity Constraints for TE Tunnels, on page 128

Assigning Color Names to Numeric Values

The first task in enabling the new coloring scheme is to assign a numerical value (in hexadecimal) to each value (color).

Note An affinity color name cannot exceed 64 characters. An affinity value cannot exceed a single digit. For example, magenta1.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **affinity-map** *affinity name* {*affinity value* | **bit-position** *value*}
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 3	**affinity-map** *affinity name* {*affinity value*	**bit-position** *value*} **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# affinity-map red 1`	Enters an affinity name and a map value by using a color name (repeat this command to assign multiple colors up to a maximum of 64 colors). An affinity color name cannot exceed 64 characters. The value you assign to a color name must be a single digit.
Step 4	**commit**	

Related Topics

Flexible Name-based Tunnel Constraints, on page 95
Configure Flexible Name-based Tunnel Constraints: Example, on page 153

Associating Affinity-Names with TE Links

The next step in the configuration of MPLS-TE Flexible Name-based Tunnel Constraints is to assign affinity names and values to TE links. You can assign up to a maximum of 32 colors. Before you assign a color to a link, you must define the name-to-value mapping for each color.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **interface** *type interface-path-id*
4. **attribute-names** *attribute name*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 3	**interface** *type interface-path-id*
	Example:
	`RP/0/RP0/CPU0:router(config-mpls-te)# interface tunnel-te 2`
	`RP/0/RP0/CPU0:router(config-mpls-te-if)#`
		Enables MPLS-TE on an interface and enters MPLS-TE interface configuration mode.

Step 4	**attribute-names** *attribute name* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te-if)# attribute-names red`	Assigns colors to TE links over the selected interface.
Step 5	**commit**	

Related Topics

Flexible Name-based Tunnel Constraints, on page 95
Configure Flexible Name-based Tunnel Constraints: Example, on page 153
Assigning Color Names to Numeric Values, on page 126

Associating Affinity Constraints for TE Tunnels

The final step in the configuration of MPLS-TE Flexible Name-based Tunnel Constraints requires that you associate a tunnel with affinity constraints.

Using this model, there are no masks. Instead, there is support for four types of affinity constraints:

- include
- include-strict
- exclude
- exclude-all

Note For the affinity constraints above, all but the exclude-all constraint may be associated with up to 10 colors.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **affinity** {*affinity-value* **mask** *mask-value* | **exclude** *name* | **exclude -all** | **include** *name* | **include-strict** *name*}
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
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	**affinity** {*affinity-value* **mask** *mask-value* | **exclude** *name* | **exclude -all** | **include** *name* | **include-strict** *name*} **Example:** `RP/0/RP0/CPU0:router(config-if)# affinity include red`	Configures link attributes for links comprising a tunnel. You can have up to ten colors. Multiple include statements can be specified under tunnel configuration. With this configuration, a link is eligible for CSPF if it has at least a red color or has at least a green color. Thus, a link with red and any other colors as well as a link with green and any additional colors meet the above constraint.
Step 4	**commit**	

Related Topics

Flexible Name-based Tunnel Constraints, on page 95

Configure Flexible Name-based Tunnel Constraints: Example, on page 153

Configuring IS-IS to Flood MPLS-TE Link Information

Perform this task to configure a router running the Intermediate System-to-Intermediate System (IS-IS) protocol to flood MPLS-TE link information into multiple IS-IS levels.

This procedure shows how to enable MPLS-TE in both IS-IS Level 1 and Level 2.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **net** *network-entity-title*
4. **address-family** {**ipv4** | **ipv6**} {**unicast**}
5. **metric-style wide**
6. **mpls traffic-eng** *level*
7. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router isis** *instance-id* **Example:** `RP/0/RP0/CPU0:router(config)# router isis 1`	Enters an IS-IS instance.
Step 3	**net** *network-entity-title* **Example:** `RP/0/RP0/CPU0:router(config-isis)# net 47.0001.0000.0000.0002.00`	Enters an IS-IS network entity title (NET) for the routing process.
Step 4	**address-family** {**ipv4**	**ipv6**} {**unicast**} **Example:** `RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast`	Enters address family configuration mode for configuring IS-IS routing that uses IPv4 and IPv6 address prefixes.
Step 5	**metric-style wide** **Example:** `RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide`	Enters the new-style type, length, and value (TLV) objects.

	Command or Action	Purpose
Step 6	**mpls traffic-eng** *level* **Example:** `RP/0/RP0/CPU0:router(config-isis-af)# mpls traffic-eng level-1-2`	Enters the required MPLS-TE level or levels.
Step 7	**commit**	

Configuring an OSPF Area of MPLS-TE

Perform this task to configure an OSPF area for MPLS-TE in both the OSPF backbone area 0 and area 1.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **mpls traffic-eng router-id** *ip-address*
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router ospf** *process-name* **Example:** `RP/0/RP0/CPU0:router(config)# router ospf 100`	Enters a name that uniquely identifies an OSPF routing process. ***process-name*** Any alphanumeric string no longer than 40 characters without spaces.
Step 3	**mpls traffic-eng router-id** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-ospf)# mpls traffic-eng router-id 192.168.70.1`	Enters the MPLS interface type. For more information, use the question mark (?) online help function.
Step 4	**area** *area-id* **Example:** `RP/0/RP0/CPU0:router(config-ospf)# area 0`	Enters an OSPF area identifier. ***area-id*** Either a decimal value or an IP address.

	Command or Action	Purpose
Step 5	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar)# interface POS 0/2/0/0`	Identifies an interface ID. For more information, use the question mark (?) online help function.
Step 6	**commit**	

Configuring Explicit Paths with ABRs Configured as Loose Addresses

Perform this task to specify an IPv4 explicit path with ABRs configured as loose addresses.

SUMMARY STEPS

1. **configure**
2. **explicit-path name** *name*
3. **index** *index-id* **next-address** [**loose**] **ipv4 unicast** *ip-address*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**explicit-path name** *name* **Example:** `RP/0/RP0/CPU0:router(config)# explicit-path name interarea1`	Enters a name for the explicit path.
Step 3	**index** *index-id* **next-address** [**loose**] **ipv4 unicast** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-expl-path)# index 1 next-address loose ipv4 unicast 10.10.10.10`	Includes an address in an IP explicit path of a tunnel.
Step 4	**commit**	

Configuring MPLS-TE Forwarding Adjacency

Perform this task to configure forwarding adjacency on a specific tunnel-te interface.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **forwarding-adjacency holdtime** *value*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface tunnel-te 1`	Enters MPLS-TE interface configuration mode.
Step 3	**forwarding-adjacency holdtime** *value* **Example:** `RP/0/RP0/CPU0:router(config-if)# forwarding-adjacency holdtime 60`	Configures forwarding adjacency using an optional specific holdtime value. By default, this value is 0 (milliseconds).
Step 4	**commit**	

Related Topics

MPLS-TE Forwarding Adjacency Benefits, on page 98
Configure Forwarding Adjacency: Example, on page 155

Configuring a Path Computation Client and Element

Perform these tasks to configure Path Comptation Client (PCC) and Path Computation Element (PCE):

- Configuring a Path Computation Client, on page 132
- Configuring a Path Computation Element Address, on page 133
- Configuring PCE Parameters, on page 134

Configuring a Path Computation Client

Perform this task to configure a TE tunnel as a PCC.

Note Only one TE-enabled IGP instance can be used at a time.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **path-option** *preference-priority* **dynamic pce**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface tunnel-te 6`	Enters MPLS-TE interface configuration mode and enables traffic engineering on a particular interface on the originating node.
Step 3	**path-option** *preference-priority* **dynamic pce** **Example:** `RP/0/RP0/CPU0:router(config-if)# path-option 1 dynamic pce`	Configures a TE tunnel as a PCC.
Step 4	**commit**	

Related Topics

Path Computation Element, on page 99
Configure PCE: Example, on page 156

Configuring a Path Computation Element Address

Perform this task to configure a PCE address.

Note Only one TE-enabled IGP instance can be used at a time.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **pce address ipv4** *address*
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls traffic-eng** **Example:** `RP/0/RP0/CPU0:router(config)# mpls traffic-eng`	Enters the MPLS-TE configuration mode.
Step 3	**pce address ipv4** *address* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce address ipv4 10.1.1.1`	Configures a PCE IPv4 address.
Step 4	**commit**	

Related Topics

Path Computation Element, on page 99
Configure PCE: Example, on page 156

Configuring PCE Parameters

Perform this task to configure PCE parameters, including a static PCE peer, periodic reoptimization timer values, and request timeout values.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **pce address ipv4** *address*
4. **pce peer ipv4** *address*
5. **pce keepalive** *interval*
6. **pce deadtimer** *value*
7. **pce reoptimize** *value*
8. **pce request-timeout** *value*
9. **pce tolerance keepalive** *value*
10. **commit**
11. **show mpls traffic-eng pce peer** [*address* | **all**]
12. **show mpls traffic-eng pce tunnels**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**mpls traffic-eng** **Example:** `RP/0/RP0/CPU0:router(config)# mpls traffic-eng`	Enters MPLS-TE configuration mode.
Step 3	**pce address ipv4** *address* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce address ipv4 10.1.1.1`	Configures a PCE IPv4 address.
Step 4	**pce peer ipv4** *address* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce peer address ipv4 10.1.1.1`	Configures a static PCE peer address. PCE peers are also discovered dynamically through OSPF or ISIS.
Step 5	**pce keepalive** *interval* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce keepalive 10`	Configures a PCEP keepalive interval. The range is from 0 to 255 seconds. When the keepalive interval is 0, the LSR does not send keepalive messages.
Step 6	**pce deadtimer** *value* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce deadtimer 50`	Configures a PCE deadtimer value. The range is from 0 to 255 seconds. When the dead interval is 0, the LSR does not timeout a PCEP session to a remote peer.
Step 7	**pce reoptimize** *value* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce reoptimize 200`	Configures a periodic reoptimization timer value. The range is from 60 to 604800 seconds. When the dead interval is 0, the LSR does not timeout a PCEP session to a remote peer.
Step 8	**pce request-timeout** *value* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# pce request-timeout 10`	Configures a PCE request-timeout. Range is from 5 to 100 seconds. PCC or PCE keeps a pending path request only for the request-timeout period.
Step 9	**pce tolerance keepalive** *value* **Example:**	Configures a PCE tolerance keepalive value (which is the minimum acceptable peer proposed keepalive).

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-mpls-te)# pce tolerance keepalive 10`	
Step 10	**commit**	
Step 11	**show mpls traffic-eng pce peer** [*address* | **all**] **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng pce peer`	Displays the PCE peer address and state.
Step 12	**show mpls traffic-eng pce tunnels** **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng pce tunnels`	Displays the status of the PCE tunnels.

Related Topics

Path Computation Element, on page 99
Configure PCE: Example, on page 156

Configuring Policy-based Tunnel Selection

Perform this task to configure policy-based tunnel selection (PBTS).

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **ipv4 unnumbered** *type interface-path-id*
4. **signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*}
5. **autoroute announce**
6. **destination** *ip-address*
7. **policy-class** {*1 - 7*} | {**default**}
8. **path-option** *preference-priority* {**explicit name** *explicit-path-name*}
9. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:**	Configures an MPLS-TE tunnel interface and enables traffic engineering on a particular interface on the originating node.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# interface tunnel-te 6`	
Step 3	**ipv4 unnumbered** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0`	Assigns a source address so that forwarding can be performed on the new tunnel.
Step 4	**signalled-bandwidth** {*bandwidth* [**class-type** *ct*] | **sub-pool** *bandwidth*} **Example:** `RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 10 class-type 1`	Configures the bandwidth required for an MPLS TE tunnel. Because the default tunnel priority is 7, tunnels use the default TE class map (namely, class-type 1, priority 7).
Step 5	**autoroute announce** **Example:** `RP/0/RP0/CPU0:router(config-if)# autoroute announce`	Enables messages that notify the neighbor nodes about the routes that are forwarding.
Step 6	**destination** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-if)# destination 10.1.1.1`	Assigns a destination address on the new tunnel. • Destination address is the remote node's MPLS-TE router ID. • Destination address is the merge point between backup and protected tunnels.
Step 7	**policy-class** {*1 - 7*} | {**default**} **Example:** `RP/0/RP0/CPU0:router(config-if)# policy-class 1`	Configures PBTS to direct traffic into specific TE tunnels or default class.
Step 8	**path-option** *preference-priority* {**explicit name** *explicit-path-name*} **Example:** `RP/0/RP0/CPU0:router(config-if)# path-option 1 explicit name backup-path`	Sets the path option to explicit with a given name (previously configured) and assigns the path ID.
Step 9	**commit**	

Related Topics

Policy-Based Tunnel Selection Functions, on page 101
Policy-Based Tunnel Selection, on page 100

Configuring the Automatic Bandwidth

Perform these tasks to configure the automatic bandwidth:

Configuring the Collection Frequency

Perform this task to configure the collection frequency. You can configure only one global collection frequency.

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng**
3. **auto-bw collect frequency** *minutes*
4. **commit**
5. **show mpls traffic-eng tunnels** [**auto-bw**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**mpls traffic-eng**
	Example:
	`RP/0/RP0/CPU0:router(config)# mpls traffic-eng`
	`RP/0/RP0/CPU0:router(config-mpls-te)#`
		Enters MPLS-TE configuration mode.

Step 3	**auto-bw collect frequency** *minutes* **Example:** `RP/0/RP0/CPU0:router(config-mpls-te)# auto-bw collect frequency 1`	Configures the automatic bandwidth collection frequency, and controls the manner in which the bandwidth for a tunnel collects output rate information; but does not adjust the tunnel bandwidth. ***minutes*** Configures the interval between automatic bandwidth adjustments in minutes. Range is from 1 to 10080.
Step 4	**commit**	
Step 5	**show mpls traffic-eng tunnels** [**auto-bw**] **Example:** `RP/0/RP0/CPU0:router# show mpls traffic tunnels auto-bw`	Displays information about MPLS-TE tunnels for the automatic bandwidth. The globally configured collection frequency is displayed.

Related Topics

MPLS-TE Automatic Bandwidth Overview, on page 102
Configure Automatic Bandwidth: Example, on page 157

Forcing the Current Application Period to Expire Immediately

Perform this task to force the current application period to expire immediately on the specified tunnel. The highest bandwidth is applied on the tunnel before waiting for the application period to end on its own.

SUMMARY STEPS

1. **mpls traffic-eng auto-bw apply** {**all** | **tunnel-te** *tunnel-number*}
2. **commit**
3. **show mpls traffic-eng tunnels** [**auto-bw**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**mpls traffic-eng auto-bw apply** {**all** | **tunnel-te** *tunnel-number*} **Example:** `RP/0/RP0/CPU0:router# mpls traffic-eng auto-bw apply tunnel-te 1`	Configures the highest bandwidth available on a tunnel without waiting for the current application period to end. **all** Configures the highest bandwidth available instantly on all the tunnels. **tunnel-te** Configures the highest bandwidth instantly to the specified tunnel. Range is from 0 to 65535.
Step 2	**commit**	
Step 3	**show mpls traffic-eng tunnels** [**auto-bw**] **Example:** `RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels auto-bw`	Displays information about MPLS-TE tunnels for the automatic bandwidth.

Configuring the Automatic Bandwidth Functions

Perform this task to configure the following automatic bandwidth functions:

Application frequency

Configures the application frequency in which a tunnel bandwidth is updated by the automatic bandwidth.

Bandwidth collection

Configures only the bandwidth collection.

Bandwidth parameters

Configures the minimum and maximum automatic bandwidth to set on a tunnel.

Adjustment threshold

Configures the adjustment threshold for each tunnel.

Overflow detection

Configures the overflow detection for each tunnel.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **auto-bw**
4. **application** *minutes*
5. **bw-limit** {**min** *bandwidth* } {**max** *bandwidth*}
6. **adjustment-threshold** *percentage* [**min** *minimum-bandwidth*]
7. **overflow threshold** *percentage* [**min** *bandwidth*] **limit** *limit*
8. **commit**
9. **show mpls traffic-eng tunnels** [**auto-bw**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	**interface tunnel-te** *tunnel-id*
	Example:
	`RP/0/RP0/CPU0:router(config)# interface tunnel-te 6`
	`RP/0/RP0/CPU0:router(config-if)#`
		Configures an MPLS-TE tunnel interface and enables traffic engineering on a particular interface on the originating node.

Step 3	**auto-bw**
	Example:
	`RP/0/RP0/CPU0:router(config-if)# auto-bw`
	`RP/0/RP0/CPU0:router(config-if-tunte-autobw)#`
		Configures automatic bandwidth on a tunnel interface and enters MPLS-TE automatic bandwidth interface configuration mode.

Step 4	**application** *minutes* **Example:** `RP/0/RP0/CPU0:router(config-if-tunte-autobw)# application 1000`	Configures the application frequency in minutes for the applicable tunnel. ***minutes*** Frequency in minutes for the automatic bandwidth application. Range is from 5 to 10080 (7 days). The default value is 1440 (24 hours).

	Command or Action	Purpose
Step 5	**bw-limit** {**min** *bandwidth* } {**max** *bandwidth*} **Example:** `RP/0/RP0/CPU0:router(config-if-tunte-autobw)# bw-limit min 30 max 80`	Configures the minimum and maximum automatic bandwidth set on a tunnel. **min** Applies the minimum automatic bandwidth in kbps on a tunnel. Range is from 0 to 4294967295. **max** Applies the maximum automatic bandwidth in kbps on a tunnel. Range is from 0 to 4294967295.
Step 6	**adjustment-threshold** *percentage* [**min** *minimum-bandwidth*] **Example:** `RP/0/RP0/CPU0:router(config-if-tunte-autobw)# adjustment-threshold 50 min 800`	Configures the tunnel bandwidth change threshold to trigger an adjustment. ***percentage*** Bandwidth change percent threshold to trigger an adjustment if the largest sample percentage is higher or lower than the current tunnel bandwidth. Range is from 1 to 100 percent. The default value is 5 percent. **min** Configures the bandwidth change value to trigger an adjustment. The tunnel bandwidth is changed only if the largest sample is higher or lower than the current tunnel bandwidth. Range is from 10 to 4294967295 kilobits per second (kbps). The default value is 10 kbps.
Step 7	**overflow threshold** *percentage* [**min** *bandwidth*] **limit** *limit* **Example:** `RP/0/RP0/CPU0:router(config-if-tunte-autobw)# overflow threshold 100 limit 1`	Configures the tunnel overflow detection. ***percentage*** Bandwidth change percent to trigger an overflow. Range is from 1 to 100 percent. **limit** Configures the number of consecutive collection intervals that exceeds the threshold. The bandwidth overflow triggers an early tunnel bandwidth update. Range is from 1 to 10 collection periods. The default value is none. **min** Configures the bandwidth change value in kbps to trigger an overflow. Range is from 10 to 4294967295. The default value is 10.
Step 8	**commit**	
Step 9	**show mpls traffic-eng tunnels** [**auto-bw**] **Example:**	Displays the MPLS-TE tunnel information only for tunnels in which the automatic bandwidth is enabled.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels auto-bw`	

Related Topics

MPLS-TE Automatic Bandwidth Overview, on page 102
Configure Automatic Bandwidth: Example, on page 157

Configuring the Shared Risk Link Groups

To activate the MPLS traffic engineering SRLG feature, you must configure the SRLG value of each link that has a shared risk with another link.

Implementing Associated Bidirectional Label Switched Paths

This section describes how to configure MPLS Traffic Engineering Associated Bidirectional Label Switched Paths (MPLS-TE LSPs).

Associated Bidirectional Label Switched Paths are LSP instances where the forward and the reverse direction paths are setup, monitored and protected independently and associated together during signaling. You use a RSVP Association object to bind the two forward and reverse LSPs together to form either a co-routed or non co-routed associated bidirectional TE tunnel.

Signaling Methods and Object Association for Bidirectional LSPs, on page 142 ,Associated Bidirectional Non Co-routed and Co-routed LSPs, on page 143 provides details.

You can associate a protecting MPLS-TE tunnel with either a working MPLS-TE LSP, protecting MPLS-TE LSP, or both. The working LSP is the primary LSP backed up by the protecting LSP. When a working LSP goes down, the protecting LSP is automatically activated. You can configure a MPLS-TE tunnel to operate without protection as well.

Path Protection, on page 147 provides details.

Signaling Methods and Object Association for Bidirectional LSPs

This section provides an overview of the association signaling methods for the bidirectional LSPs. Two unidirectional LSPs can be bound to form an associated bidirectional LSP in the following scenarios:

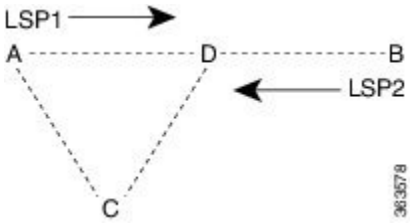
- No unidirectional LSP exists, and both must be established.
- Both unidirectional LSPs exist, but the association must be established.
- One unidirectional LSP exists, but the reverse associated LSP must be established.

Configuration information regarding the LSPs can be provided at one or both endpoints of the associated bidirectional LSP. Depending on the method chosen, there are two models of creating an associated bidirectional LSP; single-sided provisioning, and double-sided provisioning.

- **Single-sided Provisioning:** For the single-sided provisioning, the TE tunnel is configured only on one side. An LSP for this tunnel is initiated by the initiating endpoint with the Association Object inserted in the Path message. The other endpoint then creates the corresponding reverse TE tunnel and signals the reverse LSP in response to this. Currently, there is no support available for configuring single-sided provisioning.

- **Double-sided Provisioning:** For the double-sided provisioning, two unidirectional TE tunnels are configured independently on both sides. The LSPs for the tunnels are signaled with Association Objects inserted in the Path message by both sides to indicate that the two LSPs are to be associated to form a bidirectional LSP.

Consider this topology (an example of associated bidirectional LSP):



Here, LSP1 from A to B, takes the path A,D,B and LSP2 from B to A takes the path B,D,C,A. These two LSPs, once established and associated, form an associated bidirectional LSP between node A and node B. For the double sided provisioning model, both LSP1 and LSP2 are signaled independently with (Extended) Association Object inserted in the Path message, in which the Association Type indicating double-sided provisioning. In this case, the two unidirectional LSPs are bound together to form an associated bidirectional LSP based on identical Association Objects in the two LSPs' Path messages.

Association Object: An Association Object is used to bind unidirectional LSPs originating from both endpoints. The Association Object takes the following values:

- **Association Type**: In order to bind two reverse unidirectional LSPs to be an associated bidirectional LSP, the Association Type must be set to indicate either single sided or double sided LSPs.
- **Association ID**: For both single sided and double sided provisioning, Association ID must be set to a value assigned by the node that originates the association for the bidirectional LSP. This is set to the Tunnel ID of the bound LSP or the Tunnel ID of the binding LSP.
- **Association Source:** For double sided provisioning, Association Source must be set to an address selected by the node that originates the association for the bidirectional LSP. For single sided provisioning, Association Source must be set to an address assigned to the node that originates the LSP.
- **Global ID**: This is the global ID for the association global source. This must be set to the global ID of the node that originates the association for the bidirectional LSP.

Note You must provide identical values for the content of the Association Object on either end of the participating LSPs to ensure successful binding of the LSPs.

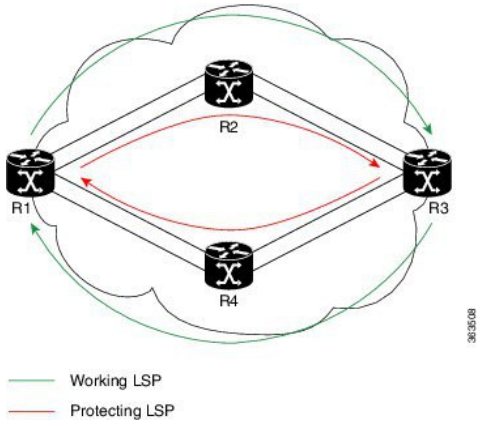
Configure Associated Bidirectional Co-routed LSPs, on page 145 describes the procedure to create associated bidirectional co-routed LSPs.

Associated Bidirectional Non Co-routed and Co-routed LSPs

This section provides an overview of associated bidirectional non co-routed and co-routed LSPs. Establishment of MPLS TE-LSP involves computation of a path between a head-end node to a tail-end node, signaling along the path, and modification of intermediate nodes along the path. The signaling process ensures bandwidth reservation (if signaled bandwidth is lesser than 0 and programming of forwarding entries.

Path computation is performed by the head-end nodes of both the participating LSPs using Constrained Shortest Path First (CSPF). CSPF is the 'shortest path (measured in terms of cost) that satisfies all relevant LSP TE constraints or attributes, such as required bandwidth, priority and so on.

Associated Bidirectional Non Co-routed LSPs: A non co-routed bidirectional TE LSP follows two different paths, that is, the forward direction LSP path is different than the reverse direction LSP path. Here is an illustration.



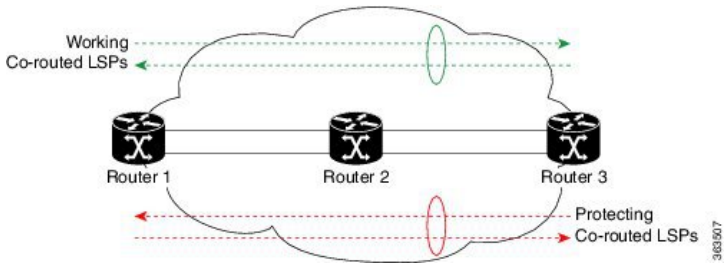
In the above topology:

- The outer paths (in green) are working LSP pairs.
- The inner paths (in red) are protecting LSP pairs.
- Router 1 sets up working LSP to Router 3 and protecting LSP to Router 3 independently.
- Router 3 sets up working LSP to Router 1 and protecting LSP to Router 1 independently.

Non co-routed bidirectional TE LSP is available by default, and no configuration is required.

Note In case of non co-routed LSPs, the head-end nodes relax the constraint on having identical forward and reverse paths. Hence, depending on network state you can have identical forward and reverse paths, though the bidirectional LSP is co-routed.

Associated Bidirectional Co-routed LSPs: A co-routed bidirectional TE LSP denotes a bidirectional tunnel where the forward direction LSP and reverse direction LSP must follow the same path, for example, the same nodes and paths. Here is an illustration.



In the above topology:

- Paths at the top of the figure (in green) indicate working co-routed LSP pairs.
- Paths at the bottom of the figure (in red) indicate protecting co-routed LSP pairs.
- Router 1 sets up working LSP to Router 3 (in red) after performing bidirectional CSPF and sends reverse explicit route object (ERO) to Router 3. Node Router 3 uses the received reverse ERO to set up reverse red working LSP to Router 1.

- Router 3 sets up protecting LSP to Router 1 (in green) after performing bidirectional CSPF and sends reverse ERO to Router 1. Node Router 1 uses the received reverse ERO to set up reverse green protecting LSP to Router 3.

Configure Associated Bidirectional Co-routed LSPs, on page 145 describes the procedure to configure an associated bidirectional co-routed LSP.

Configure Associated Bidirectional Co-routed LSPs

A co-routed bidirectional packet LSP is a combination of two LSPs (one in the forward direction and the other in reverse direction) sharing the same path between a pair of ingress and egress nodes. It is established using the extensions to RSVP-TE. This type of LSP can be used to carry any of the standard types of MPLS-based traffic, including Layer 2 VPNs, Layer 2 circuits, and Layer 3 VPNs. You can configure a single BFD session for the bidirectional LSP (that is, you do not need to configure a BFD session for each LSP in each direction). You can also configure a single standby bidirectional LSP to provide a backup for the primary bidirectional LSP.

Before you begin

- You must have symmetric source and destination TE router IDs in order for bidirectional LSPs to be associated.
- Tunnels attributes must be configured identically on both sides of co-routed bidirectional LSP.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **bidirectional**
4. **association** {**id** <0-65535> | **source-address** <IP address>} [**global-id** <0-4294967295>]
5. **association type co-routed**
6. **commit**
7. **show mpls traffic-eng tunnels bidirectional-associated co-routed**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RSP0/CPU0:router# interface tunnel-te 1`	Configures an MPLS-TE tunnel interface.
Step 3	**bidirectional** **Example:** `RP/0/0/CPU0:router(config-if)# bidirectional`	Configure the ingress router for the LSP and include the bidirectional statement to specify that the LSP be established as a bidirectional LSP.
Step 4	**association** {**id** <0-65535>	**source-address** <IP address>} [**global-id** <0-4294967295>] **Example:**	Set the association ID that uniquely identifies the association of LSPs, which is the tunnel ID of the bound LSP or the tunnel ID of the binding LSP. Also, set the source address to the tunnel sender address of the bound LSP or the tunnel

	Command or Action	Purpose
	`RP/0/0/CPU0:router(config-if-bidir)# association id 1 source-address 11.0.0.1`	sender address of the binding LSP. Optionally, specify the global ID for association global source. **Note** Association ID, association source and global ID must be configured identically on both the endpoints.
Step 5	**association type co-routed** **Example:** `RP/0/0/CPU0:router(config-if-bidir)#association type co-routed`	Specify that the LSP be established as a associated co-routed bidirectional LSP.
Step 6	**commit**	
Step 7	**show mpls traffic-eng tunnels bidirectional-associated co-routed** **Example:** `RP/0/0/CPU0:router#show mpls traffic-eng tunnels bidirectional-associated co-routed`	Shows details of an associated co-routed bidirectional LSP.

Show output for an associated co-routed bidirectional LSP configuration

This is a sample of the output for the **show mpls traffic-eng tunnels role head** command.

```
RP/0/RSP0/CPU0:router# show mpls traffic-eng tunnels role head

Name: tunnel-te1  Destination: 49.49.49.2
  Signalled-Name: IMC0_t1
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 1,  type dynamic  (Basis for Setup, path weight 20 (reverse 20))
    path option 1,  type dynamic  (Basis for Standby, path weight 20 (reverse 20))
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Sun May  4 12:09:56 2014 (03:24:11 ago)
  Config Parameters:
    Bandwidth:        0 kbps (CT0) Priority:  7  7 Affinity: 0x0/0xffff
    Metric Type: TE (default)
    Hop-limit: disabled
    Cost-limit: disabled
    AutoRoute: disabled  LockDown: disabled   Policy class: not set
    Forward class: 0 (default)
    Forwarding-Adjacency: disabled
    Loadshare:          0 equal loadshares
    Auto-bw: disabled
    Fast Reroute: Disabled, Protection Desired: None
    Path Protection: Enabled
    Association Type: Single Sided Bidirectional LSPs, Co-routed: YES
    Association ID: 100, Source: 49.49.49.2
    Reverse Bandwidth: 0 kbps (CT0), Standby: 0 kbps (CT0)
    BFD Fast Detection: Enabled
    BFD Parameters: Min-interval 100 ms (default), Multiplier 3 (default)
    BFD Bringup Timeout: Interval 60 seconds (default)
    BFD Initial Dampening: 16000 ms (default)
    BFD Maximum Dampening: 600000 ms (default)
```

```
BFD Secondary Dampening: 20000 ms (default)
Periodic LSP Ping: Interval 120 seconds (default)
Session Down Action: ACTION_REOPTIMIZE, Reopt Timeout: 300
BFD Encap Mode: GAL
Reoptimization after affinity failure: Enabled
Soft Preemption: Disabled
```

Path Protection

Path protection provides an end-to-end failure recovery mechanism (that is, full path protection) for associated bidirectional MPLS-TE LSPs. Associated bidirectional MPLS-TE LSPs support 1:1 path protection. You can configure the working and protecting LSPs as part of configuring the MPLS-TE tunnel. The working LSP is the primary LSP used to route traffic, while the protecting LSP is a backup for a working LSP. If the working LSP fails, traffic is switched to the protecting LSP until the working LSP is restored, at which time traffic forwarding reverts back to the working LSP.

When FRR is not enabled on a tunnel, and when GAL-BFD and/or Fault OAM is enabled on an associated bidirectional co-routed LSP, path-protection is activated by the FIB running on the line card that hosts the working LSP. The failure on the working LSP can be detected using BFD or Fault OAM.

Configure Path Protection for Associated Bidirectional LSPs, on page 147 provides procedural details.

You can use the **show mpls traffic-eng fast-reroute log** command to confirm whether protection switching has been activated by FIB.

Configure Path Protection for Associated Bidirectional LSPs

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **ipv4 unnumbered** *type interface-path-id*
4. **bfd** {fast-detect | encap-mode}
5. **destination** *ip-address*
6. **bidirectional**
7. **bidirectional association** {**id** <0-65535> | **source-address** <IP address>} [**global-id** <0-4294967295>
8. **association type co-routed**
9. **path-protection**
10. **path-option** *preference - priority* **{dynamic | explicit}**
11. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RSP0/CPU0:router# interface tunnel-te 1`	Configures an MPLS-TE tunnel interface.

	Command or Action	Purpose
Step 3	**ipv4 unnumbered** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0`	Assigns a source address so that forwarding can be performed on the new tunnel. Loopback is commonly used as the interface type.

Step 4	**bfd** {fast-detect | encap-mode}
	Example:
	`RP/0/RSP0/CPU0:IMC0(config-if)#bfd`
	`RP/0/RSP0/CPU0:IMC0(config-if-tunte-bfd)#fast-detect`
	`RP/0/RSP0/CPU0:IMC0(config-if-tunte-bfd)#encap-mode gal`
		Specify if you want BFD enabled for the LSP over a Generic Associated Channel (G-ACh) or over a IP channel. IP channel is the default.

Step 5	**destination** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-if)# destination 49.49.49.2`	Assigns a destination address on the new tunnel. The destination address is the remote node's MPLS-TE router ID.
Step 6	**bidirectional** **Example:** `Router(config-if)# bidirectional`	Configure the ingress router for the LSP and include the bidirectional statement to specify that the LSP be established as a bidirectional LSP.
Step 7	**bidirectional association** {**id** <0-65535> | **source-address** <IP address>} [**global-id** <0-4294967295>] **Example:** `Router(config-if-bidir)# association id 1 source-address 11.0.0.1`	Set the association ID that uniquely identifies the association of LSPs, which is the tunnel ID of the bound LSP or the tunnel ID of the binding LSP. Also, set the source address to the tunnel sender address of the bound LSP or the tunnel sender address of the binding LSP. Also, set the ID for associating the global source. **Note** Association ID, association source and optional global-id must be configured identically on both the endpoints.
Step 8	**association type co-routed** **Example:** `Router(config-if-bidir)#association type co-routed`	Specify that the LSP be established as a associated co-routed bidirectional LSP.
Step 9	**path-protection** **Example:** `RP/0/RSP0/CPU0:IMC0(config-if-bidir-co-routed)#path-protection`	Enable path protection.
Step 10	**path-option** *preference - priority* **{dynamic | explicit}** **Example:** `RP/0/RP0/CPU0:router(config-if)# path-option 1`	Sets the path option and assigns the path-option ID. Both sides of the co-routed bidirectional LSPs must use dynamic or matching co-routed strict-hop explicit path-option.

	Command or Action	Purpose
	`dynamic`	
Step 11	**commit**	

Example

Here is a sample configuration with path protection defined for the Associated Bidirectional LSP.

```
RP/0/RSP0/CPU0:IMC0#config
RP/0/RSP0/CPU0:IMC0(config)#interface tunnel-te 1
RP/0/RSP0/CPU0:IMC0(config-if)#ipv4 unnumbered loopback0
RP/0/RSP0/CPU0:IMC0(config-if)#destination 49.49.49.2
RP/0/RSP0/CPU0:IMC0(config-if)#bidirectional
RP/0/RSP0/CPU0:IMC0(config-if-bidir)#association id 100 source-address 49.49.4$
RP/0/RSP0/CPU0:IMC0(config-if-bidir)#association type co-routed
RP/0/RSP0/CPU0:IMC0(config-if-bidir-co-routed)#path-protection
RP/0/RSP0/CPU0:IMC0(config-if)#path-option 1  dynamic
RP/0/RSP0/CPU0:IMC0(config-if)#commit
```

OAM Support for Associated Bidirectional LSPs

You can opt to configure operations, administration and management (OAM) support for Associated Bidirectional LSPs in the following areas:

- **Continuity check:** You can configure bidirectional forwarding detection (BFD) over a Generic Associated Channel (G-ACh) with hardware assist. This allows for BFD Hello packets to be generated and processed in hardware making smaller Hello intervals such as 3.3 ms feasible. For more information on BFD and BFD hardware offload see *Implementing BFD* module in the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide* .
- **Fault notification:** You can run Fault OAM over associated bidirectional co-routed LSPs to convey fault notification from mid-point to end-point of the LSP. The following fault OAM messages are supported:
  - Link Down Indication (LDI): generated when an interface goes down (for example, to fiber-cut) at mid-point.
  - Lock Report (LKR): generated when an interface is shutdown at mid-point.

  You can configure fault OAM to generate OAM message at mid-point or enable protection switching due to fault OAM at end-point. Generate Fault OAM Messages at Mid-point, on page 149 and Generate Fault OAM Messages at End-point, on page 150provides procedural details.
- **Fault diagnostics:** You can use the ping and traceroute features as a means to check connectivity and isolate failure points for both co-routed and non-co-routed bidirectional TE tunnels. *MPLS Network Management with MPLS LSP Ping and MPLS SP Traceroute* provides details.

Generate Fault OAM Messages at Mid-point

To program all bi-directional LSPs to generate fault OAM message at mid-point use the following steps:

SUMMARY STEPS

1. **configure**

2. **mpls traffic-eng**
3. **fault-oam**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**mpls traffic-eng** **Example:** `RP/0/RSP0/CPU0:IMO(config)# mpls traffic-eng`	Configures an MPLS-TE tunnel interface.
Step 3	**fault-oam** **Example:** `RP/0/RSP0/CPU0:IMC0(config-mpls-te)#fault-oam`	Enable fault OAM for an associated bidirectional LSP.
Step 4	**commit**	

Generate Fault OAM Messages at End-point

In order to enable protection switching due to fault OAM at end-point use the following steps:

SUMMARY STEPS

1. **configure**
2. **interface tunnel-te** *tunnel-id*
3. **bidirectional association type co-routed fault-oam**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface tunnel-te** *tunnel-id* **Example:** `RP/0/RSP0/CPU0:IMC0(config)#interface tunnel-te 1`	Configures an MPLS-TE tunnel interface.
Step 3	**bidirectional association type co-routed fault-oam** **Example:** `RP/0/RSP0/CPU0:IMC0(config-if)#bidirectional association type co-routed fault-oam`	Enable fault OAM for an associated co-routed bidirectional LSP.
Step 4	**commit**	

Pseudowire Call Admission Control

You can use the Pseudowire Call Admission Control (PW CAC) process to check for bandwidth constraints and ensure that once the path is signaled, the links (pseudowires) participating in the bidirectional LSP association have the required bandwidth. Only pseudowires with sufficient bandwidth are admitted in the bidirectional LSP association process. *Configure Pseudowire Bandwidth* in the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* provides procedural details.

Configuration Examples for Cisco MPLS-TE

These configuration examples are used for MPLS-TE:

Build MPLS-TE Topology and Tunnels: Example

The following examples show how to build an OSPF and IS-IS topology:

```
(OSPF)
...
configure
  mpls traffic-eng
  interface pos 0/6/0/0
  router id loopback 0
  router ospf 1
  router-id 192.168.25.66
  area 0
  interface pos 0/6/0/0
  interface loopback 0
  mpls traffic-eng router-id 192.168.70.1
  mpls traffic-eng area 0
  rsvp
  interface pos 0/6/0/0
  bandwidth 100
  commit
show mpls traffic-eng topology
show mpls traffic-eng link-management advertisement
!
(IS-IS)
...
configure
  mpls traffic-eng
  interface pos 0/6/0/0
  router id loopback 0
  router isis lab
  address-family ipv4 unicast
  mpls traffic-eng level 2
  mpls traffic-eng router-id 192.168.70.2
  !
  interface POS0/0/0/0
  address-family ipv4 unicast
!
```

The following example shows how to configure tunnel interfaces:

```
interface tunnel-te1
  destination 192.168.92.125
```

```
  ipv4 unnumbered loopback 0
  path-option l dynamic
  bandwidth 100
  commit
show mpls traffic-eng tunnels
show ipv4 interface brief
show mpls traffic-eng link-management admission-control
!
interface tunnel-te1
  autoroute announce
  route ipv4 192.168.12.52/32 tunnel-te1
  commit
ping 192.168.12.52
show mpls traffic autoroute
!
interface tunnel-te1
  fast-reroute
  mpls traffic-eng interface pos 0/6/0/0
  backup-path tunnel-te 2
  interface tunnel-te2
  backup-bw global-pool 5000
  ipv4 unnumbered loopback 0
  path-option l explicit name backup-path
  destination 192.168.92.125
 commit
show mpls traffic-eng tunnels backup
show mpls traffic-eng fast-reroute database
!
rsvp
  interface pos 0/6/0/0
  bandwidth 100 150 sub-pool 50
  interface tunnel-te1
  bandwidth sub-pool 10
commit
```

Related Topics

Building MPLS-TE Topology, on page 108
Creating an MPLS-TE Tunnel, on page 110
How MPLS-TE Works, on page 89

Configure IETF DS-TE Tunnels: Example

The following example shows how to configure DS-TE:

```
rsvp
 interface pos 0/6/0/0
 bandwidth rdm 100 150 bc1 50
 mpls traffic-eng
 ds-te mode ietf
 interface tunnel-te 1
 bandwidth 10 class-type 1
 commit

configure
 rsvp interface 0/6/0/0
 bandwidth mam max-reservable-bw 400 bc0 300 bc1 200
 mpls traffic-eng
 ds-te mode ietf
 ds-te model mam
 interface tunnel-te 1bandwidth 10 class-type 1
```

```
commit
```

Related Topics

Configuring a Prestandard DS-TE Tunnel, on page 117
Prestandard DS-TE Mode, on page 90

Configure MPLS-TE and Fast-Reroute on OSPF: Example

CSPF areas are configured on a per-path-option basis. The following example shows how to use the traffic-engineering tunnels (tunnel-te) interface and the active path for the MPLS-TE tunnel:

```
configure
 interface tunnel-te 0
  path-option 1 explicit id 6 ospf 126 area 0
  path-option 2 explicit name 234 ospf 3 area 7 verbatim
  path-option 3 dynamic isis mtbf level 1 lockdown
  commit
```

Configure the Ignore IS-IS Overload Bit Setting in MPLS-TE: Example

This example shows how to configure the IS-IS overload bit setting in MPLS-TE:

```
configure
 mpls traffic-eng
  path-selection ignore overload
   commit
```

Related Topics

Configuring the Ignore Integrated IS-IS Overload Bit Setting in MPLS-TE, on page 125
Ignore Intermediate System-to-Intermediate System Overload Bit Setting in MPLS-TE, on page 94

Configure Flexible Name-based Tunnel Constraints: Example

The following configuration shows the three-step process used to configure flexible name-based tunnel constraints.

```
R2
line console
 exec-timeout 0 0
 width 250
!
logging console debugging
explicit-path name mypath
 index 1 next-address loose ipv4 unicast 3.3.3.3 !
explicit-path name ex_path1
 index 10 next-address loose ipv4 unicast 2.2.2.2  index 20 next-address loose ipv4 unicast
3.3.3.3 !
interface Loopback0
 ipv4 address 22.22.22.22 255.255.255.255 !
```

```
interface tunnel-te1
 ipv4 unnumbered Loopback0
 signalled-bandwidth 1000000
 destination 3.3.3.3
 affinity include green
 affinity include yellow
 affinity exclude indigo
 affinity exclude orange
 path-option 1 dynamic
!
router isis 1
 is-type level-1
 net 47.0001.0000.0000.0001.00
 nsf cisco
 address-family ipv4 unicast
  metric-style wide
  mpls traffic-eng level-1
  mpls traffic-eng router-id 192.168.70.1
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/1/0/0
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/1/0/1
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/1/0/2
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/1/0/3
  address-family ipv4 unicast
  !
 !
!
rsvp
 interface GigabitEthernet0/1/0/0
  bandwidth 1000000 1000000
 !
 interface GigabitEthernet0/1/0/1
  bandwidth 1000000 1000000
 !
 interface GigabitEthernet0/1/0/2
  bandwidth 1000000 1000000
 !
 interface GigabitEthernet0/1/0/3
  bandwidth 1000000 1000000
 !
!
mpls traffic-eng
 interface GigabitEthernet0/1/0/0
  attribute-names red purple
 !
 interface GigabitEthernet0/1/0/1
  attribute-names red orange
 !
 interface GigabitEthernet0/1/0/2
  attribute-names green purple
```

```
 !
 interface GigabitEthernet0/1/0/3
  attribute-names green orange
 !
 affinity-map red 1
 affinity-map blue 2
 affinity-map teal 80
 affinity-map green 4
 affinity-map indigo 40
 affinity-map orange 20
 affinity-map purple 10
 affinity-map yellow 8
!
```

Related Topics

Assigning Color Names to Numeric Values, on page 126
Associating Affinity-Names with TE Links, on page 127
Associating Affinity Constraints for TE Tunnels, on page 128
Flexible Name-based Tunnel Constraints, on page 95

Configure an Interarea Tunnel: Example

The following configuration example shows how to configure a traffic engineering interarea tunnel. .

Note Specifying the tunnel tailend in the loosely routed path is optional.

```
configure
  interface Tunnel-te1
    ipv4 unnumbered Loopback0
    destination 192.168.20.20
    signalled-bandwidth 300
    path-option 1 explicit name path-tunnel1

explicit-path name path-tunnel1
  index 10 next-address loose ipv4 unicast 192.168.40.40
  index 20 next-address loose ipv4 unicast 192.168.60.60
  index 30 next-address loose ipv4 unicast 192.168.20.20
```

Configure Forwarding Adjacency: Example

The following configuration example shows how to configure an MPLS-TE forwarding adjacency on tunnel-te 68 with a holdtime value of 60:

```
configure
 interface tunnel-te 68
 forwarding-adjacency holdtime 60
 commit
```

Related Topics

Configuring MPLS-TE Forwarding Adjacency, on page 131

MPLS-TE Forwarding Adjacency Benefits, on page 98

Configure PCE: Example

The following configuration example illustrates a PCE configuration:

```
configure
mpls traffic-eng
  interface pos 0/6/0/0
  pce address ipv4 192.168.25.66
  router id loopback 0
  router ospf 1
  router-id 192.168.25.66
  area 0
  interface pos 0/6/0/0
  interface loopback 0
  mpls traffic-eng router-id 192.168.70.1
  mpls traffic-eng area 0
  rsvp
  interface pos 0/6/0/0
  bandwidth 100
  commit
```

The following configuration example illustrates PCC configuration:

```
configure
  interface tunnel-te 10
  ipv4 unnumbered loopback 0
  destination 1.2.3.4
  path-option 1 dynamic pce
  mpls traffic-eng
  interface pos 0/6/0/0
  router id loopback 0
  router ospf 1
  router-id 192.168.25.66
  area 0
  interface pos 0/6/0/0
  interface loopback 0
  mpls traffic-eng router-id 192.168.70.1
  mpls traffic-eng area 0
  rsvp
  interface pos 0/6/0/0
  bandwidth 100
  commit
```

Related Topics

Configuring a Path Computation Client, on page 132
Configuring a Path Computation Element Address, on page 133
Configuring PCE Parameters, on page 134
Path Computation Element, on page 99

Configure Policy-based Tunnel Selection: Example

The following configuration example illustrates a PBTS configuration:

```
configure
 interface tunnel-te0
 ipv4 unnumbered Loopback3
 signalled-bandwidth 50000
 autoroute announce
 destination 1.5.177.2
 policy-class 2
 path-option 1 dynamic
```

Configure Automatic Bandwidth: Example

The following configuration example illustrates an automatic bandwidth configuration:

```
configure
 interface tunnel-te6
  auto-bw
   bw-limit min 10000 max 500000
   overflow threshold 50 min 1000 limit 3
   adjustment-threshold 20 min 1000
   application 180
```

Related Topics

Configuring the Collection Frequency, on page 138
Configuring the Automatic Bandwidth Functions, on page 139
MPLS-TE Automatic Bandwidth Overview, on page 102

Configure Entropy Labels for MPLS TE Networks

Most MPLS networks use load balancing techniques for traffic engineering. What causes latency in such widespread networks is the time taken to inspect the label stack at each transit Label Switching Router (LSR) to determine the next hop or path.

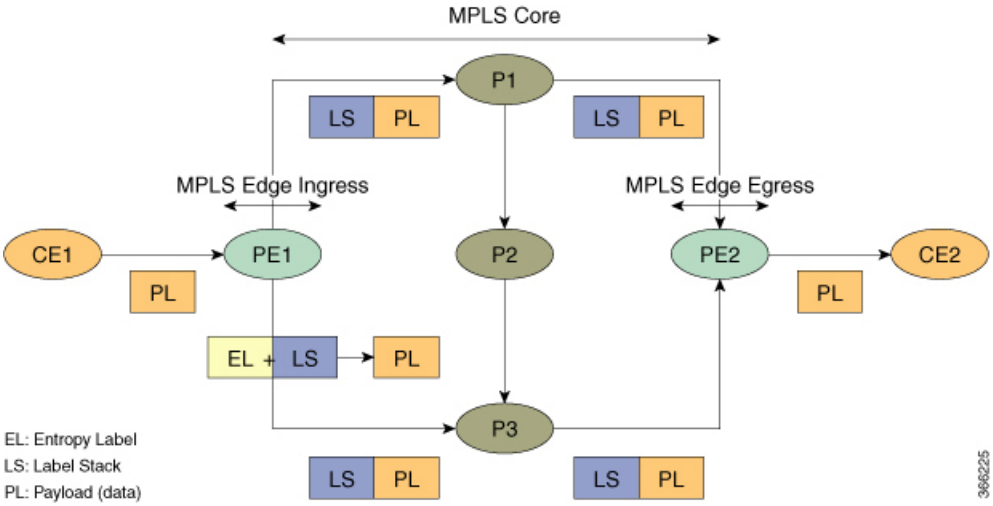
The latency can be reduced by inserting a label known as the *entropy label* on top of the label stack at the ingress LSR. The entropy label contains the keys required by the load balancing function, and thus eliminates the need for deep packet inspection at transit LSRs. The ingress LSR, which has all the information about incoming packets, extracts the load balancing keys from the entropy label and decides the optimum paths for the packets. The transit LSRs use the rest of the label stack to forward the packets along the pre-determined paths.

The advantages of using entropy labels in MPLS networks are:

- Ingress LSRs operate at lower bandwidths than transit LSRs, and are hence the ideal choice for load balancing.
- Transit LSRs do not need to perform deep packet inspection and can effectively load balance the packets as decided by the Ingress LSRs.
- Transit LSRs are spared from the problem of misinterpreting the protocol denoted in the label stack and thereby causing inequitable distribution of traffic across equal cost paths exiting from the LSR.

The following illustration shows the transit of a packet through the MPLS network. The entropy label is attached at the ingress router for load balancing. When the optimum path is determined for the packet, which contains the payload (data) and the label stack, the entropy label is no longer required.

Figure 13: Transit of an MPLS Packet with an Entropy Label



Configuration

1. To configure an MPLS entropy label, use the following configuration.

```
RP/0/RP0/CPU0:router(config)# mpls ldp
RP/0/RP0/CPU0:router(config-ldp)# entropy-label
RP/0/RP0/CPU0:router(config-ldp)# commit
RP/0/RP0/CPU0:router(config-ldp)# end
```

2. Locate the route that needs to use the entropy label for load balancing.

```
RP/0/RP0/CPU0:router# show cef exact-route 10.1.6.1 10.1.1.1

10.1.1.1/32, version 40, internal 0x1000001 0x0 (ptr 0x8d42b4d8) [1], 0x0 (0x8d5c5020),
 0xa20 (0x8e1c0098)
...
Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via Bundle-Ether613
   via 11.1.5.1/32, Bundle-Ether613, 2 dependencies, weight 0, class 0 [flags 0x0]
    path-idx 2 NHID 0x0 [0x8dd02920 0x8dd02810]
    next hop 11.1.5.1/32
    local adjacency
     local label 24002      labels imposed {ImplNull}
```

3. Use the route to pass the entropy label for load balancing.

   You are prompted for the option of entering the entropy label for multiple source-destination pairs.

```
RP/0/RP0/CPU0:router# bundle-hash bundle-Ether 613
Specify load-balance configuration (L3/3-tuple or L4/7-tuple) (L3,L4): L3
Single SA/DA pair (IPv4,IPv6) or range (IPv4 only) or Entropy Label (MPLS only): S/R/E
[S]: E

Enter Entropy Label(in network byte order): 14001

Entropy Label 14001 -- Link hashed to is TenGigE0/1/0/8/8
```

```
Another? [y]:
```

4. Verify if traffic is getting load balanced with the MPLS entropy label configuration.

```
RP/0/RP0/CPU0:router# show mpls forwarding exact-route label 24002 entropy-label 14001

Local  Outgoing    Prefix            Outgoing     Next Hop        Bytes
Label  Label       or ID             Interface                    Switched
------ ----------- ----------------- ------------ --------------- ------------
24002  24010    10.1.1.1/32        BE613        11.1.5.1/32       N/A
     Via: BE613, Next Hop: 11.1.5.1/32
     Label Stack (Top -> Bottom): { 24010 }
     NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx: 0, Weight: 0
     MAC/Encaps: 0/4, MTU: 1500
```

You have successfully configured an MPLS entropy label in your network.

Additional References

For additional information related to implementing MPLS-TE, refer to the following references:

Related Documents

Related Topic	Document Title
MPLS-TE commands	*MPLS Traffic Engineering Commands* module in *MPLS Command Reference for Cisco NCS 6000 Series Routers*.

Standards

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

MIBs

MIBs	MIBs Link
—	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

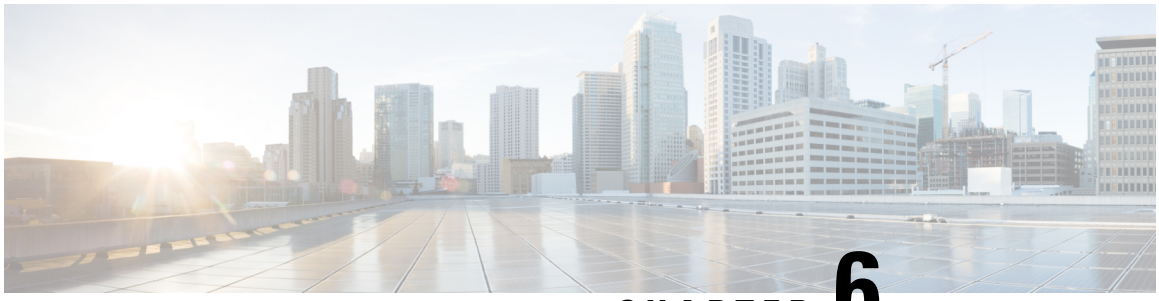
RFCs

RFCs	Title
RFC 4124	*Protocol Extensions for Support of Diffserv-aware MPLS Traffic Engineering*, F. Le Faucheur, Ed. June 2005. (Format: TXT=79265 bytes) (Status: PROPOSED STANDARD)

RFCs	Title
RFC 4125	*Maximum Allocation Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*, F. Le Faucheur, W. Lai. June 2005. (Format: TXT=22585 bytes) (Status: EXPERIMENTAL)
RFC 4127	*Russian Dolls Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*, F. Le Faucheur, Ed. June 2005. (Format: TXT=23694 bytes) (Status: EXPERIMENTAL)

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



CHAPTER 6

Implementing MPLS OAM

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.

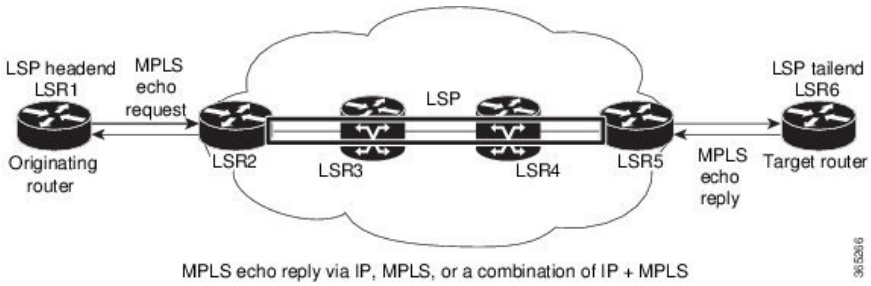
MPLS LSP Ping

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 14: MPLS LSP Ping Echo Request and Reply Paths



By default, the **ping mpls ipv4** command tries to determine the Forwarding Equivalence Class (FEC) being used automatically. However, this is only applicable at head-end and works only if the FEC at the destination is same as the source. If the source and destination FEC types are not the same, the **ping mpls ipv4** command may fail to identify the targeted FEC type. You can overcome this limitation by specifying the FEC type in MPLS LSP ping using the **fec-type** command option. If the user is not sure about the FEC type at the transit or the destination, or it may change through network, use of the **generic** FEC type command option is recommended. Generic FEC is not coupled to a particular control plane and allows path verification when the advertising protocol is unknown, or may change during the path of the echo request. If you are aware of the destination FEC type, specify the target FEC as BGP or LDP.

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 address.

```
RP/0/RP0/CPU0:router# ping mpls ipv4 10.1.1.2/32 verbose

Sun Nov 15 11:27:43.070 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.

!      size 100, reply addr 10.1.0.2, return code 3
!      size 100, reply addr 10.1.0.2, return code 3
!      size 100, reply addr 10.1.0.2, return code 3
!      size 100, reply addr 10.1.0.2, return code 3
!      size 100, reply addr 10.1.0.2, return code 3

Success rate is 100 percent (5/5), round-trip min/avg/max = 2/2/4 ms
```

In this example, 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
```

In this example, the destination is specified as a Label Distribution Protocol (LDP) IPv4 prefix and the FEC type is specified as BGP.

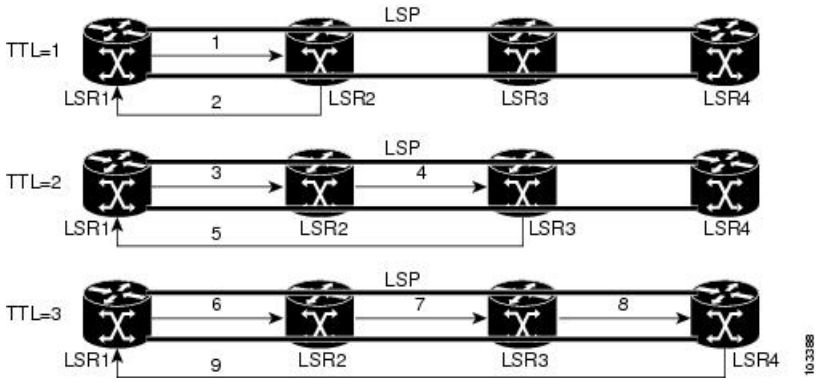
```
RP/0/RP0/CPU0:router# ping mpls ipv4 10.1.1.2/32 fec-type bgp

Wed Nov 25 03:38:33.143 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
```

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.

The following figure shows an MPLS LSP traceroute example with an LSP from LSR1 to LSR4.

Figure 15: MPLS LSP Traceroute



By default, the **traceroute mpls ipv4** command tries to determine the Forwarding Equivalence Class (FEC) being used automatically. However, this is only applicable at head-end and works only if the FEC at the destination is same as the source. If the source and destination FEC types are not the same, the **traceroute mpls ipv4** command may fail to identify the targeted FEC type. You can overcome this limitation by specifying the FEC type in MPLS LSP traceroute using the **fec-type** command option. If the user is not sure about the FEC type at the transit or the destination, or it may change through network, use of the **generic** FEC type command option is recommended. Generic FEC is not coupled to a particular control plane and allows path verification when the advertising protocol is unknown, or may change during the path of the echo request. If you are aware of the destination FEC type, specify the target FEC as BGP or LDP.

Configuration Examples

This example shows how to use the **traceroute** command to trace to a destination.

```
RP/0/RP0/CPU0:router# traceroute mpls ipv4 10.1.1.2/32 destination 127.0.0.3 127.0.0.6 2
Sat Jan 27 03:50:23.746 UTC

Tracing MPLS Label Switched Path to 10.1.1.2/32, 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.

Destination address 127.0.0.3
  0 10.2.1.2 MRU 1500 [Labels: 24000 Exp: 0]
L 1 10.2.1.1 MRU 1500 [Labels: implicit-null Exp: 0] 8 ms
! 2 10.1.0.2 3 ms

Destination address 127.0.0.5
  0 10.2.1.2 MRU 1500 [Labels: 24000 Exp: 0]
L 1 10.2.1.1 MRU 1500 [Labels: implicit-null Exp: 0] 5 ms
! 2 10.1.0.2 2 ms
```

This example shows how to use the **traceroute** command and how to specify the maximum number of hops for the traceroute to traverse by specifying the **ttl** value.

```
RP/0/RP0/CPU0:router# traceroute mpls ipv4 10.1.1.2/32 ttl 1
Sun Nov 15 12:20:14.145 UTC
Tracing MPLS Label Switched Path to 10.1.1.2/32, 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 10.1.0.1 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.1.0.2 3 ms
```

This example shows how to use the **traceroute** command to trace to a destination and FEC type is specified as generic.

```
RP/0/RP0/CPU0:router# traceroute mpls ipv4 10.1.1.2/32 fec-type generic
Sun Nov 15 12:25:14.145 UTC
Tracing MPLS Label Switched Path to 10.1.1.2/32, 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 10.12.12.1 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.12.12.2 2 ms
```

This example shows how to use the **traceroute** command to trace to a destination and FEC type is specified as BGP.

```
RP/0/RP0/CPU0:router# traceroute mpls ipv4 10.1.1.2/32 fec-type bgp
Sun Nov 15 12:25:14.145 UTC
Tracing MPLS Label Switched Path to 10.1.1.2/32, 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 10.12.12.1 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.12.12.2 2 ms
```

Overview of P2MP TE Network

A Point to Multipoint (P2MP) TE network contains the following elements:

- *Headend Router*

  The headend router, also called the source or ingress router, is responsible for initiating the signaling messages that set up the P2MP TE LSP. The headend router can also be a branch point, which means the router performs packet replication and the sub-LSPs split into different directions.

- *Midpoint Router*

  The midpoint router is where the sub-LSP signaling is processed. The midpoint router can be a branch point.

- *Tailend Router*

  The tailend router, also called the destination, egress, or leaf-node router, is where sub-LSP signaling ends. The router which is one of potentially many destinations of the P2MP TE LSP.

- *Bud Router*

  A bud router is a midpoint and tailend router at the same time. An LSR that is an egress LSR, but also has one or more directly connected downstream LSRs.

- *Branch Router*

  A branch router is either a midpoint or tailend router at any given time.

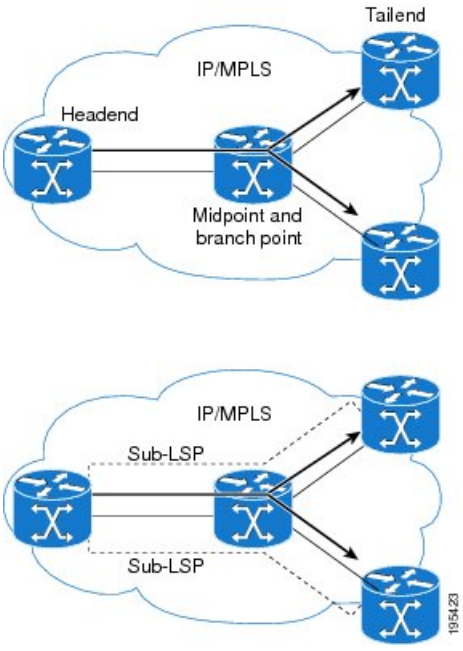
- *Transit Router*

  A transit router is an LSR that is not an egress router, but also has one or more directly connected downstream routers.

- A P2MP tunnel consists of one or more sub-LSPs.All sub-LSPs belonging to the same P2MP tunnel employ the same constraints, protection policies, and so on, which are configured at the headend router.

Figure 16: Elements of P2MP TE Network illustrates the elements of P2MP TE network.

Figure 16: Elements of P2MP TE Network



P2MP TE tunnels build on the features that exist in basic point-to-point TE tunnels. The P2MP TE tunnels have the following characteristics:

- There is one source (headend) but more than one destination (tailend).
- They are unidirectional.

- They are explicitly routed.
- Multiple sub-LSPs connect the headend router to various tailend routers.

P2MP Ping

The P2MP ping feature is used to check the connectivity between Ingress LSR and egress LSR, along a P2MP LSP. The Ingress LSR sends the P2MP echo request message along the specified P2MP LSP. All egress LSRs which receive the P2MP echo request message from the ingress LSR must send a P2MP echo reply message to the ingress LSR, according to the reply mode specified in the P2MP echo request message.

P2MP Traceroute

The P2MP traceroute feature is used to isolate the failure point of a P2MP LSP.

Traceroute can be applied to all nodes in the P2MP tree. However, you can select a specific traceroute target through the P2MP Responder Identifier TLV. An entry in this TLV represents an responder-id or a transit node. This is only the case for P2MP TE LSPs.

Note Only P2MP TE LSP IPv4 is supported. If the Responder Identifier TLV is missing, the **echo request** requests information from all responder-ids.

MPLS OAM Support for BGP 3107

The MPLS OAM Support for BGP 3107 feature provides support for ping, traceroute and treetrace (traceroute multipath) operations for LSPs signaled via BGP for the IPv4 unicast prefix FECs in the default VRF, according to the *RFC 3107 - Carrying Label Information in BGP-4*. This feature adds support for MPLS OAM operations in the seamless MPLS architecture deployments, i.e., combinations of BGP and LDP signaled LSPs.

Configuration Examples: P2MP Ping and P2MP Traceroute

This example shows an extract of the P2MP ping command.

```
RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-mte 10
Sending 1, 100-byte MPLS Echos to tunnel-mte10,
      timeout is 2.2 seconds, send interval is 0 msec, jitter value is 200 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, 'd' - DDMAP

Type escape sequence to abort.

Request #1
! reply addr 192.168.222.2
! reply addr 192.168.140.2
! reply addr 192.168.170.1
```

```
Success rate is 100 percent (3 received replies/3 expected replies),
     round-trip min/avg/max = 154/232/302 ms
```

This example shows an extract of the P2MP ping command with the jitter option.

```
RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-mte 10 jitter 300

Sending 1, 100-byte MPLS Echos to tunnel-mte10,
      timeout is 2.3 seconds, send interval is 0 msec, jitter value is 300 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, 'd' - DDMAP

Type escape sequence to abort.

Request #1
! reply addr 192.168.222.2
! reply addr 192.168.140.2
! reply addr 192.168.170.1

Success rate is 100 percent (3 received replies/3 expected replies),
     round-trip min/avg/max = 148/191/256 ms
```

This example shows an extract of the P2MP ping command with the ddmap option.

```
RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-mte 10 ddmap

Sending 1, 100-byte MPLS Echos to tunnel-mte10,
      timeout is 2.2 seconds, send interval is 0 msec, jitter value is 200 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, 'd' - DDMAP

Type escape sequence to abort.

Request #1
! reply addr 192.168.222.2
! reply addr 192.168.140.2
! reply addr 192.168.170.1

Success rate is 100 percent (3 received replies/3 expected replies),
     round-trip min/avg/max = 105/178/237 ms


RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels p2mp 10
Mon Apr 12 12:13:55.075 EST
Signalling Summary:
              LSP Tunnels Process:  running
```

```
                      RSVP Process:  running
                        Forwarding:  enabled
              Periodic reoptimization:  every 3600 seconds, next in 654 seconds
               Periodic FRR Promotion:  every 300 seconds, next in 70 seconds
              Auto-bw enabled tunnels:  0 (disabled)

Name: tunnel-mte10
   Status:
     Admin: up  Oper: up (Up for 12w4d)

     Config Parameters:
      Bandwidth: 0 kbps (CT0) Priority: 7 7 Affinity: 0x0/0xffff
      Metric Type: TE (default)
      Fast Reroute: Not Enabled, Protection Desired: None
      Record Route: Not Enabled

      Destination summary: (3 up, 0 down, 0 disabled) Affinity: 0x0/0xffff
      Auto-bw: disabled
      Destination: 10.1.0.1
        State: Up for 12w4d
        Path options:
          path-option 1 dynamic     [active]
      Destination: 10.2.0.1
        State: Up for 12w4d
        Path options:
          path-option 1 dynamic     [active]
      Destination: 10.3.0.1
        State: Up for 12w4d
        Path options:
          path-option 1 dynamic     [active]

     History:
       Reopt. LSP:
         Last Failure:
           LSP not signalled, identical to the [CURRENT] LSP
           Date/Time: Thu Jan 14 02:49:22 EST 2010 [12w4d ago]

    Current LSP:
      lsp-id: 10002 p2mp-id: 10 tun-id: 10 src: 10.0.0.1 extid: 10.0.0.1
      LSP up for: 12w4d
      Reroute Pending: No
      Inuse Bandwidth: 0 kbps (CT0)
      Number of S2Ls: 3 connected, 0 signaling proceeding, 0 down

      S2L Sub LSP: Destination 10.1.0.1 Signaling Status: connected
        S2L up for: 12w4d
        Sub Group ID: 1 Sub Group Originator ID: 10.1.0.1
        Path option path-option 1 dynamic    (path weight 1)
        Path info (OSPF 1 area 0)
          192.168.222.2
          10.1.0.1

      S2L Sub LSP: Destination 10.2.0.1 Signaling Status: connected
        S2L up for: 12w4d
        Sub Group ID: 2 Sub Group Originator ID: 10.0.0.1
        Path option path-option 1 dynamic    (path weight 2)
        Path info (OSPF 1 area 0)
          192.168.222.2
          192.168.140.3
          192.168.140.2
          10.2.0.1

      S2L Sub LSP: Destination 10.3.0.1 Signaling Status: connected
        S2L up for: 12w4d
```

```
        Sub Group ID: 3 Sub Group Originator ID: 10.0.0.1
        Path option path-option 1 dynamic    (path weight 2)
        Path info (OSPF 1 area 0)
          192.168.222.2
          192.168.170.3
          192.168.170.1
          10.3.0.1

    Reoptimized LSP (Install Timer Remaining 0 Seconds):
      None
    Cleaned LSP (Cleanup Timer Remaining 0 Seconds):
      None
Displayed 1 (of 16) heads, 0 (of 0) midpoints, 0 (of 0) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads

RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-mte 10 lsp id 10002
Mon Apr 12 12:14:04.532 EST

Sending 1, 100-byte MPLS Echos to tunnel-mte10,
      timeout is 2.2 seconds, send interval is 0 msec, jitter value is 200 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, 'd' - DDMAP

Type escape sequence to abort.

Request #1
! reply addr 192.168.222.2
! reply addr 192.168.170.1
! reply addr 192.168.140.2

Success rate is 100 percent (3 received replies/3 expected replies),
     round-trip min/avg/max = 128/153/167 ms
```

This example shows an extract of the P2MP ping command with the responder-id.

```
RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-mte 10 responder-id 10.3.0.1
Mon Apr 12 12:15:34.205 EST

Sending 1, 100-byte MPLS Echos to tunnel-mte10,
      timeout is 2.2 seconds, send interval is 0 msec, jitter value is 200 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, 'd' - DDMAP

Type escape sequence to abort.

Request #1
! reply addr 192.168.170.1

Success rate is 100 percent (1 received reply/1 expected reply),
     round-trip min/avg/max = 179/179/179 ms
```

This example shows an extract of the P2MP traceroute command with the ttl option.

```
RP/0/RP0/CPU0:router# traceroute mpls traffic-eng tunnel-mte 10  ttl 4
Mon Apr 12 12:16:50.095 EST

Tracing MPLS MTE Label Switched Path on tunnel-mte10, timeout is 2.2 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, 'd' - DDMAP

Type escape sequence to abort.

! 1 192.168.222.2 186 ms [Estimated Role: Bud]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]

! 2 192.168.222.2 115 ms [Estimated Role: Bud]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 2 192.168.140.2 213 ms [Estimated Role: Egress]
! 2 192.168.170.1 254 ms [Estimated Role: Egress]

! 3 192.168.222.2 108 ms [Estimated Role: Bud]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 3 192.168.170.1 164 ms [Estimated Role: Egress]
! 3 192.168.140.2 199 ms [Estimated Role: Egress]

! 4 192.168.170.1 198 ms [Estimated Role: Egress]
! 4 192.168.222.2 206 ms [Estimated Role: Bud]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500
```

This example shows an extract of the P2MP traceroute command with the responder-id option.

```
RP/0/RP0/CPU0:router# traceroute mpls traffic-eng tunnel-mte 10 responder-id 10.3.0.1
Mon Apr 12 12:18:01.994 EST

Tracing MPLS MTE Label Switched Path on tunnel-mte10, timeout is 2.2 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, 'd' - DDMAP

Type escape sequence to abort.

d 1 192.168.222.2 113 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]

d 2 192.168.222.2 118 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
```

```
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 2 192.168.170.1 244 ms [Estimated Role: Egress]

d 3 192.168.222.2 141 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 3 192.168.170.1 204 ms [Estimated Role: Egress]

d 4 192.168.222.2 110 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 4 192.168.170.1 174 ms [Estimated Role: Egress]
```

This example shows an extract of the P2MP traceroute command with the jitter option.

```
RP/0/RP0/CPU0:router# traceroute mpls traffic-eng tunnel-mte 10 responder-id 10.3.0.1 ttl
4 jitter 500
Mon Apr 12 12:19:00.292 EST

Tracing MPLS MTE Label Switched Path on tunnel-mte10, timeout is 2.5 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, 'd' - DDMAP

Type escape sequence to abort.

d 1 192.168.222.2 238 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]

d 2 192.168.222.2 188 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 2 192.168.170.1 290 ms [Estimated Role: Egress]

d 3 192.168.222.2 115 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 3 192.168.170.1 428 ms [Estimated Role: Egress]

d 4 192.168.222.2 127 ms [Estimated Role: Branch]
    [L] DDMAP 0: 192.168.140.2 192.168.140.2 MRU 1500 [Labels: 16001 Exp: 0]
    [L] DDMAP 1: 192.168.170.1 192.168.170.1 MRU 1500 [Labels: 16000 Exp: 0]
! 4 192.168.170.1 327 ms [Estimated Role: Egress]
```