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

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

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

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS” WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE. IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

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

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

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

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

© 2018–2019 Cisco Systems, Inc. All rights reserved.
CONTENTS

PREFACE

Preface vii
  Changes to This Document vii
  Communications, Services, and Additional Information vii

CHAPTER 1
New and Changed Information for Segment Routing Features 1
  New and Changed Segment Routing Features 1

CHAPTER 2
About Segment Routing 3
  Scope 3
  Need 4
  Benefits 4
  Workflow for Deploying Segment Routing 5

CHAPTER 3
Configure Segment Routing Global Block and Segment Routing Local Block 7
  About the Segment Routing Global Block 7
  About the Segment Routing Local Block 8
  Setup a Non-Default Segment Routing Global Block Range 9
  Setup a Non-Default Segment Routing Local Block Range 10

CHAPTER 4
Configure Segment Routing for IS-IS Protocol 13
  Enabling Segment Routing for IS-IS Protocol 13
  Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface 16
  Configuring an Adjacency SID 18
    Configuring a Layer 2 Adjacency SID 21
  Configuring Bandwidth-Based Local UCMP 23
  IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability 25
Contents

CHAPTER 5

Configure Segment Routing for OSPF Protocol 31

Enabling Segment Routing for OSPF Protocol 31

Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface 33

CHAPTER 6

Configure Segment Routing for BGP 37

Segment Routing for BGP 37

Configure BGP Prefix Segment Identifiers 38

Configure Segment Routing Egress Peer Engineering 39

Configure BGP Link-State 40

Example: Configuring SR-EPE and BGP-LS 41

Configure BGP Proxy Prefix SID 43

CHAPTER 7

Configure SR-TE Policies 47

Configure SR-TE Policies 47

Autoroute Include 50

Color-Only Steering 50

Address-Family Agnostic Steering 51

Flexible Name-based Policy Constraints 52

Configuring Flexible Name-Based Policy Constraints 52

Affinity Support for Anycast SIDs 54

BGP SR-TE 57

Configure Explicit BGP SR-TE 57

Setting CO Flag 60

Using Binding Segments 61

Stitching SR-TE Polices Using Binding SID: Example 62
CHAPTER 8 Configure Segment Routing Path Computation Element 65
  About SR-PCE 65
  Configure SR-PCE 66
    Configure the Disjoint Policy (Optional) 68
  PCE-Initiated SR Policies for Traffic Management 69

CHAPTER 9 Configure Topology-Independent Loop-Free Alternate (TI-LFA) 73
  Configuring TI-LFA for IS-IS 73
  Configuring TI-LFA for OSPF 75
  TI-LFA Node and SRLG Protection: Examples 77
  Configuring Global Weighted SRLG Protection 78

CHAPTER 10 Configure Segment Routing Microloop Avoidance 81
  About Segment Routing Microloop Avoidance 81
  Configure Segment Routing Microloop Avoidance for IS-IS 81
  Configure Segment Routing Microloop Avoidance for OSPF 82

CHAPTER 11 Configure Segment Routing Mapping Server 85
  Segment Routing Mapping Server 85
    Segment Routing Mapping Server Restrictions 86
  Segment Routing and LDP Interoperability 86
    Example: Segment Routing LDP Interoperability 86
  Configuring Mapping Server 88
  Enable Mapping Advertisement 90
    Configure Mapping Advertisement for IS-IS 90
    Configure Mapping Advertisement for OSPF 91
  Enable Mapping Client 92

CHAPTER 12 Using Segment Routing Traffic Matrix 93
  Segment Routing Traffic Matrix 93
  Traffic Collector Process 93
  Configuring Traffic Collector 94
  Displaying Traffic Information 95
CHAPTER 13 Using Segment Routing OAM 97
MPLS Ping and Traceroute for BGP and IGP Prefix-SID 97
Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID 98
MPLS LSP Ping and Traceroute Nil FEC Target 100
Examples: LSP Ping and Traceroute for Nil_FEC Target 100
Segment Routing Ping 101
Segment Routing Traceroute 104
Segment Routing Policy Nil-FEC Ping and Traceroute 106
Preface

The Cisco ASR 9000 Series Aggregation Services Segment Router Routing Configuration Guide preface contains these sections:

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

Changes to This Document

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

<table>
<thead>
<tr>
<th>Date</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>July 2018</td>
<td>Initial release of this document</td>
</tr>
<tr>
<td>January 2019</td>
<td>Republished for Cisco IOS XR Release 6.5.2</td>
</tr>
<tr>
<td>March 2019</td>
<td>Republished for Cisco IOS XR Release 6.5.3</td>
</tr>
</tbody>
</table>

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.
# New and Changed Information for Segment Routing Features

This table summarizes the new and changed feature information for the *Segment Routing Configuration Guide for Cisco ASR 9000 Aggregation Services Routers*, and lists where they are documented.

- New and Changed Segment Routing Features, on page 1

## New and Changed Segment Routing Features

### Segment Routing Features Added or Modified in IOS XR Release 6.5.x

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Introduced/Changed in Release</th>
<th>Where Documented</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCE-Initiated Segment Routing Policies</td>
<td>This feature is introduced.</td>
<td>Release 6.5.1</td>
<td>PCE-Initiated SR Policies for Traffic Management, on page 69</td>
</tr>
<tr>
<td>SR-TE Policy with Autoroute Include</td>
<td>This feature is introduced.</td>
<td>Release 6.5.1</td>
<td>Autoroute Include, on page 50</td>
</tr>
<tr>
<td>Color-Only Steering</td>
<td>This feature is introduced.</td>
<td>Release 6.5.1</td>
<td>Color-Only Steering, on page 50</td>
</tr>
<tr>
<td>Address-Family Agnostic Steering</td>
<td>This feature is introduced.</td>
<td>Release 6.5.1</td>
<td>Address-Family Agnostic Steering, on page 51</td>
</tr>
</tbody>
</table>
About Segment Routing

Segment Routing is not supported on 1st generation Cisco ASR 9000 Ethernet Line Cards or the Cisco ASR 9000 SIP-700 SPA Interface Processor. Refer to the Cisco ASR 9000 Ethernet Line Card Installation Guide for details about 1st generation line cards.

This chapter introduces the concept of segment routing and provides a workflow for configuring segment routing.

- Scope, on page 3
- Need, on page 4
- Benefits, on page 4
- Workflow for Deploying Segment Routing, on page 5

Scope

Segment routing is a method of forwarding packets on the network based on the source routing paradigm. The source chooses a path and encodes it in the packet header as an ordered list of segments. Segments are an identifier for any type of instruction. For example, topology segments identify the next hop toward a destination. Each segment is identified by the segment ID (SID) consisting of a flat unsigned 20-bit integer.

Segments

Interior gateway protocol (IGP) distributes two types of segments: prefix segments and adjacency segments. Each router (node) and each link (adjacency) has an associated segment identifier (SID).

- A prefix SID is associated with an IP prefix. The prefix SID is manually configured from the segment routing global block (SRGB) range of labels, and is distributed by IS-IS or OSPF. The prefix segment steers the traffic along the shortest path to its destination. A node SID is a special type of prefix SID that identifies a specific node. It is configured under the loopback interface with the loopback address of the node as the prefix.

A prefix segment is a global segment, so a prefix SID is globally unique within the segment routing domain.

- An adjacency segment is identified by a label called an adjacency SID, which represents a specific adjacency, such as egress interface, to a neighboring router. An adjacency SID can be allocated dynamically from the dynamic label range or configured manually from the segment routing local block (SRLB) range...
of labels. The adjacency SID is distributed by IS-IS or OSPF. The adjacency segment steers the traffic to a specific adjacency.

An adjacency segment is a local segment, so the adjacency SID is locally unique relative to a specific router.

By combining prefix (node) and adjacency segment IDs in an ordered list, any path within a network can be constructed. At each hop, the top segment is used to identify the next hop. Segments are stacked in order at the top of the packet header. When the top segment contains the identity of another node, the receiving node uses equal cost multipaths (ECMP) to move the packet to the next hop. When the identity is that of the receiving node, the node pops the top segment and performs the task required by the next segment.

Dataplane
Segment routing can be directly applied to the Multiprotocol Label Switching (MPLS) architecture with no change in the forwarding plane. A segment is encoded as an MPLS label. An ordered list of segments is encoded as a stack of labels. The segment to process is on the top of the stack. The related label is popped from the stack, after the completion of a segment.

Services
Segment Routing integrates with the rich multi-service capabilities of MPLS, including Layer 3 VPN (L3VPN), Virtual Private Wire Service (VPWS), Virtual Private LAN Service (VPLS), and Ethernet VPN (EVPN).

Segment Routing for Traffic Engineering
Segment routing for traffic engineering (SR-TE) takes place through a tunnel between a source and destination pair. Segment routing for traffic engineering uses the concept of source routing, where the source calculates the path and encodes it in the packet header as a segment. Each segment is an end-to-end path from the source to the destination, and instructs the routers in the provider core network to follow the specified path instead of the shortest path calculated by the IGP. The destination is unaware of the presence of the tunnel.

Need
With segment routing for traffic engineering (SR-TE), the network no longer needs to maintain a per-application and per-flow state. Instead, it simply obeys the forwarding instructions provided in the packet.

SR-TE utilizes network bandwidth more effectively than traditional MPLS-TE networks by using ECMP at every segment level. It uses a single intelligent source and relieves remaining routers from the task of calculating the required path through the network.

Benefits

- **Ready for SDN**: Segment routing was built for SDN and is the foundation for Application Engineered Routing (AER). SR prepares networks for business models, where applications can direct network behavior. SR provides the right balance between distributed intelligence and centralized optimization and programming.

- **Minimal configuration**: Segment routing for TE requires minimal configuration on the source router.
- **Load balancing**: Unlike in RSVP-TE, load balancing for segment routing can take place in the presence of equal cost multiple paths (ECMPs).

- **Supports Fast Reroute (FRR)**: Fast reroute enables the activation of a pre-configured backup path within 50 milliseconds of path failure.

- **Plug-and-Play deployment**: Segment routing tunnels are interoperable with existing MPLS control and data planes and can be implemented in an existing deployment.

## Workflow for Deploying Segment Routing

Follow this workflow to deploy segment routing.

1. Configure the Segment Routing Global Block (SRGB)
2. Enable Segment Routing and Node SID on the IGP
3. Configure Segment Routing on the BGP
4. Configure the SR-TE Policy
5. Configure TI-LFA
6. Configure the Segment Routing Mapping Server
7. Collect Traffic Statistics
Chapter 3

Configure Segment Routing Global Block and Segment Routing Local Block

Local label allocation is managed by the label switching database (LSD). The Segment Routing Global Block (SRGB) and Segment Routing Local Block (SRLB) are label values preserved for segment routing in the LSD.

- About the Segment Routing Global Block, on page 7
- About the Segment Routing Local Block, on page 8
- Setup a Non-Default Segment Routing Global Block Range, on page 9
- Setup a Non-Default Segment Routing Local Block Range, on page 10

About the Segment Routing Global Block

The SRGB label values are assigned as prefix segment identifiers (SIDs) to SR-enabled nodes and have global significance throughout the domain.

Note

Because the values assigned from the range have domain-wide significance, we recommend that all routers within the domain be configured with the same range of values.

The default SRGB range is from 16000 to 23999.

Note

On SR-capable routers, the default starting value of the dynamic label range is increased from 16000 to 24000, so that the default SRGB label values (16000 to 23999) are available when SR is enabled on a running system. If a dynamic label range has been configured with a starting value of 16000, then the default SRGB label values may already be in use when SR is enabled on a running system. Therefore, you must reload the router after enabling SR to release the currently allocated labels and allocate the SRGB.

Also, if you need to increase the SRGB range after you have enabled SR, you must reload the router to release the currently allocated labels and allocate the new SRGB.

To keep the segment routing configuration simple and to make it easier to troubleshoot segment routing issues, we recommend that you use the default SRGB range on each node in the domain. However, there are instances when you might need to define a different range. For example:
• The nodes of another vendor support a label range that is different from the default SRGB, and you want to use the same SRGB on all nodes.
• The default range is too small.
• To specify separate SRGBs for IS-IS and OSPF protocols, as long as the ranges do not overlap.

Restrictions:
• In Cisco IOS XR release 6.2.x and earlier, LSD label values 0-15999 are reserved. In Cisco IOS XR release 6.3.1 and later, LSD label values 0-14999 are reserved.
• In Cisco IOS XR release 6.2.x and earlier, the maximum SRGB size is 65536. In Cisco IOS XR release 6.3.1 and later, the maximum SRGB size is 262,143.
• The SRGB upper bound cannot exceed the platform's capability.

Note
Label values that are not previously reserved are available for dynamic assignment.

The SRGB can be disabled if SR is not used.

About the Segment Routing Local Block

The Segment Routing Local Block (SRLB) is a range of label values preserved for the manual allocation of adjacency segment identifiers (adj-SIDs), Layer 2 adj-SIDs, and binding SIDs (BSIDs). These labels are locally significant and are only valid on the nodes that allocate the labels. The default SRLB range is from 15000 to 15999.

Note
Adjacency SIDs, Layer 2 adjacency SIDs, and binding SIDs (BSIDs) that are not manually allocated using the SRLB will be dynamically allocated from the dynamic label range.

To keep the segment routing configuration simple and to make it easier to troubleshoot segment routing issues, we recommend that you use the default SRLB range. However, there are instances when you might need to define a different range. For example:

• The nodes of another vendor support a label range that is different from the default SRLB, and you want to use the same SRLB on all nodes.
• The default range is too small.

When you define a new SRLB range, there might be a label conflict (for example, if labels are already allocated, statically or dynamically, in the new SRLB range). In this case, the new SRLB range will be accepted, but not applied (pending). The previous SRLB range (active) will continue to be in use until one of the following occurs:

• Reload the router to release the currently allocated labels and allocate the new SRLB.
• Use the clear segment-routing local-block discrepancy all command to clear the label conflicts.
Restrictions:
- LSD label values 0-14999 are reserved.
- The SRLB size cannot be more than 262,143.
- The SRLB upper bound cannot exceed the platform's capability.

Note
The SRLB (Segment Routing Local Block) inconsistency and allocation failure error is observed when a
non-default values of SRLB and SRGB (Segment Routing Global Block) are configured and a commit-replace
is followed by configuration re-application. This issue impacts data forwarding as the SR labels are not properly
programmed.

To prevent the issue, use the `clear segment-routing local-block discrepancy all` command to clear the label
conflicts.

The SRLB can be disabled if SR is not used.

Setup a Non-Default Segment Routing Global Block Range

This task explains how to configure a non-default SRGB range.

SUMMARY STEPS
1. configure
2. [router {isis instance-id | ospf process_name} ]
3. segment-routing global-block starting_value ending_value
4. commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 [router {isis instance-id</td>
<td>ospf process_name} ] Example: RP/0/RSP0/CPU0:router(config)# router isis 1</td>
</tr>
<tr>
<td>Step 3 segment-routing global-block starting_value ending_value Example: RP/0/RSP0/CPU0:router(config-isis)# segment-routing global-block 18000 19999</td>
<td>Enter the lowest value that you want the SRGB range to include as the starting value. Enter the highest value that you want the SRGB range to include as the ending value.</td>
</tr>
<tr>
<td>Step 4 commit</td>
<td></td>
</tr>
</tbody>
</table>
Verify the SRGB configuration:

```
RP/0/RSP0/CPU0:router# show mpls label table detail
Table Label    Owner           State    Rewrite
----- ------- ------------------------------- ------ -------
<...snip...>
0 18000 ISIS(A):1 InUse No
  Lbl-blk SRGB, vers:0, (start_label=18000, size=2000)
0 24000 ISIS(A):1 InUse Yes
  (SR Adj Segment IPv4, vers:0, index=1, type=0, intf=G10/0/0/0, nh=10.0.0.2)
```

**What to do next**

Configure prefix SIDs and enable segment routing.

---

**Setup a Non-Default Segment Routing Local Block Range**

This task explains how to configure a non-default SRLB range.

**SUMMARY STEPS**

1. configure
2. `segment-routing local-block` starting_value ending_value
3. commit

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 <code>segment-routing local-block</code> starting_value ending_value</td>
<td>Enter the lowest value that you want the SRLB range to include as the starting value. Enter the highest value that you want the SRLB range to include as the ending value.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# segment-routing local-block 30000 30999</td>
<td></td>
</tr>
</tbody>
</table>

Step 3 commit

Verify the SRLB configuration:

```
RP/0/RSP0/CPU0:router# show mpls label table detail
Table Label    Owner           State    Rewrite
----- ------- ------------------------------- ------ -------
<...snip...>
0 13 LSD(A)    InUse Yes
0 30000 LSD(A) InUse No
  Lbl-blk SRLB, vers:0, (start_label=30000, size=1000, app_notify=0)
```
Display and resolve any SRLB inconsistencies:

RP/0/RSP0/CPU0:router# `show segment-routing local-block inconsistencies`
Tue Aug 15 13:53:30.555 EDT
SRLB inconsistencies range: **Start/End: 30000/30009**

RP/0/RSP0/CPU0:router# `show mpls lsd private | i SRLB`
Tue Aug 15 13:53:50.874 EDT
SRLB Lbl Mgr:
  - Current Active SRLB block = [15000, 15999]
  - Configured Pending SRLB block = [30000, 30009]

RP/0/RSP0/CPU0:router# `clear segment-routing local-block discrepancy all`
Tue Aug 15 13:59:46.897 EDT

RP/0/RSP0/CPU0:router# `show mpls lsd private | i SRLB`
Tue Aug 15 13:59:55.370 EDT
SRLB Lbl Mgr:
  - Current Active SRLB block = [30000, 30009]
  - Configured Pending SRLB block = [0, 0]

RP/0/RSP0/CPU0:router# `show mpls label table detail private`
Tue Aug 15 14:00:26.023 EDT
Table Label Owner State Rewrite
----- ------- ------------------------------- ------ -------
0  0 LSD(A) InUse Yes
0  1 LSD(A) InUse Yes
0  2 LSD(A) InUse Yes
0 13 LSD(A) InUse Yes
0 30000 LSD(A) InUse No
(Lbl-blk SRLB, vers:0, (start_label=30000, size=1000, app_notify=0)

**What to do next**

Configure adjacency SIDs and enable segment routing.
Setup a Non-Default Segment Routing Local Block Range
CHAPTER 4

Configure Segment Routing for IS-IS Protocol


This module provides the configuration information used to enable segment routing for IS-IS.

For additional information on implementing IS-IS on your Cisco ASR 9000 Series Router, see the Implementing IS-IS module in the Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide.

- Enabling Segment Routing for IS-IS Protocol, on page 13
- Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 16
- Configuring an Adjacency SID, on page 18
- Configuring Bandwidth-Based Local UCMP, on page 23
- IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability, on page 25
- IS-IS Multi-Domain Prefix SID and Domain Stitching: Example, on page 28

Enabling Segment Routing for IS-IS Protocol

Segment routing on the IS-IS control plane supports the following:

- IPv4 and IPv6 control plane
- Level 1, level 2, and multi-level routing
- Prefix SIDs for host prefixes on loopback interfaces
- Multiple IS-IS instances on the same loopback interface for domain border nodes
- Adjacency SIDs for adjacencies
- MPLS penultimate hop popping (PHP) and explicit-null signaling

This task explains how to enable segment routing for IS-IS.
**Before you begin**

Your network must support the MPLS Cisco IOS XR software feature before you enable segment routing for IS-IS on your router.

---

**Note**

You must enter the commands in the following task list on every IS-IS router in the traffic-engineered portion of your network.

---

**SUMMARY STEPS**

1. `configure`
2. `router isis instance-id`
3. `address-family { ipv4 | ipv6 } [ unicast ]`
4. `metric-style wide [ level { 1 | 2 } ]`
5. `mpls traffic-eng level`
6. `mpls traffic-eng router-id interface`
7. `router-id loopback loopback interface used for prefix-sid`
8. `segment-routing mpls`
9. `exit`
10. `mpls traffic-eng`
11. `commit`

---

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
</tbody>
</table>
| **Step 2** | `router isis instance-id`  
**Example:**  
RP/0/RSP0/CPU0:router(config)# `router isis isp`  
| Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.  
**Note** You can change the level of routing to be performed by a particular routing instance by using the `is-type` router configuration command. |
| **Step 3** | `address-family { ipv4 | ipv6 } [ unicast ]`  
**Example:**  
RP/0/RSP0/CPU0:router(config-isis)# `address-family ipv4 unicast`  
| Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |
| **Step 4** | `metric-style wide [ level { 1 | 2 } ]`  
**Example:**  
RP/0/RSP0/CPU0:router(config-isis-af)# `metric-style wide level 1`  
| Configures a router to generate and accept only wide link metrics in the Level 1 area. |
| **Step 5** | `mpls traffic-eng level`  
**Example:**  
| Enables RSVP traffic engineering functionality. |
### Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.5.x

**Enabling Segment Routing for IS-IS Protocol**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mpls traffic-eng level-2-only</code></td>
<td>Enables traffic engineering functionality on the node. The node advertises the traffic engineering link attributes in IGP which populates the traffic engineering database (TED) on the head-end. The RSVP-TE head-end requires the TED to calculate and validate the path of the RSVP-TE policy.</td>
</tr>
</tbody>
</table>

**Step 6**

**mpls traffic-eng router-id interface**

*Example:*

```
RP/0/RSP0/CPU0:router(config-isis-af)# mpls traffic-eng router-id Loopback0
```

Sets the traffic engineering loopback interface.

**Step 7**

**router-id loopback loopback interface used for prefix-sid**

*Example:*

```
RP/0/(config-isis-af)#router-id loopback0
```

Configures router ID for each address-family (ipv4/ipv6).

**Step 8**

**segment-routing mpls**

*Example:*

```
RP/0/RSP0/CPU0:router(config-isis-af)# segment-routing mpls
```

Segment routing is enabled by the following actions:

- MPLS forwarding is enabled on all interfaces where IS-IS is active.
- All known prefix-SIDs in the forwarding plain are programmed, with the prefix-SIDs advertised by remote routers or learned through local or remote mapping server.
- The prefix-SIDs locally configured are advertised.

**Step 9**

**exit**

*Example:*

```
RP/0/RSP0/CPU0:router(config-isis-af)# exit
```

Enables traffic engineering functionality on the node.

**Step 10**

**mpls traffic-eng**

*Example:*

```
RP/0/RSP0/CPU0:router(config)# mpls traffic-eng
```

What to do next

Configure the prefix SID.
Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface

A prefix SID is associated with an IP prefix. The prefix SID is manually configured from the segment routing global block (SRGB) range of labels. The prefix segment steers the traffic along the shortest path to its destination. A node SID is a special type of prefix SID that identifies a specific node. It is configured under the loopback interface with the loopback address of the node as the prefix.

Strict-SPF SIDs are used to forward traffic strictly along the SPF path. Strict-SPF SIDs are not forwarded to SR-TE tunnels. IS-IS advertises the SR Algorithm sub Type Length Value (TLV) (in the SR Router Capability SubTLV) to include both algorithm 0 (SPF) and algorithm 1 (Strict-SPF). When the IS-IS area or level is Strict-SPF TE-capable, Strict-SPF SIDs are used to build the SR-TE Strict-SPF tunnels. Strict-SPF SIDs are also used to program the backup paths for prefixes, node SIDs, and adjacency SIDs.

Note

The same SRGB is used for both regular SIDs and strict-SPF SIDs.

The prefix SID is globally unique within the segment routing domain.

This task explains how to configure prefix segment identifier (SID) index or absolute value on the IS-IS enabled Loopback interface.

Before you begin

Ensure that segment routing is enabled on the corresponding address family.

SUMMARY STEPS

1. `configure`
2. `router isis instance-id`
3. `interface Loopback instance`
4. `address-family { ipv4 | ipv6 } [ unicast ]`
5. `prefix-sid [ strict-spf ] { index SID-index | absolute SID-value } [ n-flag-clear ] [ explicit-null ]`
6. `commit`

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><code>router isis instance-id</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config)# router isis 1</code></td>
</tr>
</tbody>
</table>
### Purpose

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 3</strong> interface Loopback instance</td>
<td>Specifies the loopback interface and instance.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis)# interface Loopback0</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> address-family { ipv4</td>
<td>ipv6 } [ unicast ]</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>The following is an example for ipv4 address family:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> prefix-sid [ strict-spf ] { index SID-index</td>
<td>absolute SID-value } [ n-flag-clear ] [ explicit-null ]</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Specify strict-spf to configure the prefix-SID to use the SPF path instead of the SR-TE tunnel.</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis-if-af)# prefix-sid index 1001</td>
<td>Specify index SID-index for each node to create a prefix SID based on the lower boundary of the SRGB + the index.</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis-if-af)# prefix-sid strict-spf index 101</td>
<td>Specify absolute SID-value for each node to create a specific prefix SID within the SRGB.</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis-if-af)# prefix-sid absolute 17001</td>
<td>By default, the n-flag is set on the prefix-SID, indicating that it is a node SID. For specific prefix-SID (for example, Anycast prefix-SID), enter the n-flag-clear keyword. IS-IS does not set the n flag in the prefix-SID sub Type Length Value (TLV).</td>
</tr>
<tr>
<td><strong>Step 6</strong> commit</td>
<td>To disable penultimate-hop-popping (PHP) and add explicit-Null label, enter explicit-null keyword. IS-IS sets the e flag in the prefix-SID sub TLV.</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>IS-IS does not advertise separate explicit-NULL or flags for regular SIDs and strict-SPF SIDs. The settings in the regular SID are used if the settings are different.</td>
</tr>
</tbody>
</table>

Verify the prefix-SID configuration:

```
RP/0/RSP0/CPU0:router# show isis database verbose
IS-IS 1 (Level-2) Link State Database
LSPID: router.00-00  LSP Seq Num: 0x0000039b  LSP Checksum: 0xfc27  LSP Holdtime: 1079 ATT/P/OL: 0/0/0
Area Address: 49.0001
NLPID: 0xcc
NLPID: 0x8e
MT: Standard (IPv4 Unicast)
MT: IPv6 Unicast
```
Configuring an Adjacency SID

An adjacency SID (Adj-SID) is associated with an adjacency to a neighboring node. The adjacency SID steers the traffic to a specific adjacency. Adjacency SIDs have local significance and are only valid on the node that allocates them.

An adjacency SID can be allocated dynamically from the dynamic label range or configured manually from the segment routing local block (SRLB) range of labels.

Adjacency SIDs that are dynamically allocated do not require any special configuration, however there are some limitations:

- A dynamically allocated Adj-SID value is not known until it has been allocated, and a controller will not know the Adj-SID value until the information is flooded by the IGP.
- Dynamically allocated Adj-SIDs are not persistent and can be reallocated after a reload or a process restart.
- Each link is allocated a unique Adj-SID, so the same Adj-SID cannot be shared by multiple links.

Manually allocated Adj-SIDs are persistent over reloads and restarts. They can be provisioned for multiple adjacencies to the same neighbor or to different neighbors. You can specify that the Adj-SID is protected. If the Adj-SID is protected on the primary interface and a backup path is available, a backup path is installed. By default, manual Adj-SIDs are not protected.

Adjacency SIDs are advertised using the existing IS-IS Adj-SID sub-TLV. The S and P flags are defined for manually allocated Adj-SIDs.

```
0 1 2 3 4 5 6 7
+aaaaaaaaa
|F|B|V|L|S|P|
+--------------
```
Table 1: Adjacency Segment Identifier (Adj-SID) Flags Sub-TLV Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S (Set)</td>
<td>This flag is set if the same Adj-SID value has been provisioned on multiple interfaces.</td>
</tr>
<tr>
<td>P (Persistent)</td>
<td>This flag is set if the Adj-SID is persistent (manually allocated).</td>
</tr>
</tbody>
</table>

Manually allocated Adj-SIDs are supported on point-to-point (P2P) interfaces.

This task explains how to configure an Adj-SID on an interface.

Before you begin

Ensure that segment routing is enabled on the corresponding address family.

Use the `show mpls label table detail` command to verify the SRLB range.

**SUMMARY STEPS**

1. `configure`
2. `router isis instance-id`
3. `interface type interface-path-id`
4. `point-to-point`
5. `address-family { ipv4 | ipv6 } [ unicast ]`
6. `adjacency-sid {index adj-SID-index | absolute adj-SID-value } [protected ]`
7. `commit`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>configure</code></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><code>router isis instance-id</code></td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:router(config)# router isis 1</code></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><code>interface type interface-path-id</code></td>
<td>Specifies the interface and enters interface configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:router(config-isis)# interface GigabitEthernet0/0/0/7</code></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>point-to-point</code></td>
<td>Specifies the interface is a point-to-point interface.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:router(config-isis-if)# point-to-point</code></td>
<td></td>
</tr>
</tbody>
</table>
### Configuring an Adjacency SID

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 5</strong> address-family {ipv4</td>
<td>ipv6} [unicast]</td>
</tr>
</tbody>
</table>

**Example:**
The following is an example for ipv4 address family:

```
RP/0/RSP0/CPU0:router(config-isis-if)#
address-family ipv4 unicast
```

| **Step 6** adjacency-sid \{index adj-SID-index | absolute adj-SID-value\} [protected] | Configures the Adj-SID index or absolute value for the interface. |

**Example:**
```
RP/0/RSP0/CPU0:router(config-isis-if-af)#
adjacency-sid index 10
```
```
RP/0/RSP0/CPU0:router(config-isis-if-af)#
adjacency-sid absolute 15010
```

<table>
<thead>
<tr>
<th><strong>Step 7</strong> commit</th>
<th></th>
</tr>
</thead>
</table>

**Verify the Adj-SID configuration:**
```
RP/0/RSP0/CPU0:router# show isis segment-routing label adjacency persistent
Mon Jun 12 02:44:07.085 PDT
IS-IS 1 Manual Adjacency SID Table
15010 AF IPv4
    GigabitEthernet0/0/0/3: IPv4, Protected 1/65/N, Active
    GigabitEthernet0/0/0/7: IPv4, Protected 2/66/N, Active
15100 AF IPv6
    GigabitEthernet0/0/0/3: IPv6, Not protected 255/255/N, Active
```

**Verify the labels are added to the MPLS Forwarding Information Base (LFIB):**
```
RP/0/RSP0/CPU0:router# show mpls forwarding labels 15010
Mon Jun 12 02:50:12.172 PDT
Local Outgoing Prefix Outgoing Interface Next Hop Bytes Switched
------ ----------- ------------------ ------------ --------------- ------------
15010 Pop SRLB (idx 10) G10/0/0/3 10.0.3.3 0
15010 Pop SRLB (idx 10) G10/0/0/7 10.1.0.5 0 (!)
15010 Pop SRLB (idx 10) G10/0/0/7 10.1.0.5 0 (!)
```

---

**Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.5.x**

20
What to do next

Configure the SR-TE policy.

Configuring a Layer 2 Adjacency SID

Typically, an adjacency SID (Adj-SID) is associated with a Layer 3 adjacency to a neighboring node, to steer the traffic to a specific adjacency. If you have Layer 2 bundle interfaces, where multiple physical interfaces form a bundle interface, the individual Layer 2 bundle members are not visible to IGP; only the bundle interface is visible.

You can configure a Layer 2 Adj-SID for the individual Layer 2 bundle interfaces. This configuration allows you to track the availability of individual bundle member links and to verify the segment routing forwarding over the individual bundle member links, for Operational Administration and Maintenance (OAM) purposes.

A Layer 2 Adj-SID can be allocated dynamically or configured manually.

- IGP dynamically allocates Layer 2 Adj-SIDs from the dynamic label range for each Layer 2 bundle member. A dynamic Layer 2 Adj-SID is not persistent and can be reallocated as the Layer 2 bundle link goes up and down.

- Manually configured Layer 2 Adj-SIDs are persistent if the Layer 2 bundle link goes up and down. Layer 2 Adj-SIDs are allocated from the Segment Routing Local Block (SRLB) range of labels. However, if the configured value of Layer 2 Adj-SID does not fall within the available SRLB, a Layer 2 Adj-SID will not be programmed into forwarding information base (FIB).

Restrictions

- Adj-SID forwarding requires a next-hop, which can be either an IPv4 address or an IPv6 address, but not both. Therefore, manually configured Layer 2 Adj-SIDs are configured per address-family.

- Manually configured Layer 2 Adj-SID can be associated with only one Layer 2 bundle member link.

- A SID value used for Layer 2 Adj-SID cannot be shared with Layer 3 Adj-SID.

- SR-TE using Layer 2 Adj-SID is not supported.

This task explains how to configure a Layer 2 Adj-SID on an interface.

Before you begin

Ensure that segment routing is enabled on the corresponding address family.

Use the show mpls label table detail command to verify the SRLB range.

SUMMARY STEPS

1. configure
2. segment-routing
3. adjacency-sid
4. interface type interface-path-id
5. address-family {ipv4 | ipv6} [unicast]
6. l2-adjacency sid {index adj-SID-index | absolute adj-SID-value} [next-hop {ipv4_address | ipv6_address}]
7. commit
### Configuring a Layer 2 Adjacency SID

8. `end`
9. `router isis instance-id`
10. `address-family { ipv4 | ipv6 } [ unicast ]`
11. `segment-routing bundle-member-adj-sid`

#### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><code>segment-routing</code>&lt;br&gt;Example: <code>Router(config)# segment-routing</code></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><code>adjacency-sid</code>&lt;br&gt;Example: <code>Router(config-sr)# adjacency-sid</code></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><code>interface type interface-path-id</code>&lt;br&gt;Example: <code>Router(config-sr-adj)# interface GigabitEthernet0/0/0/3</code></td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td>`address-family { ipv4</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td>`l2-adjacency sid {index adj-SID-index</td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td><code>commit</code></td>
</tr>
</tbody>
</table>

**Purpose**

- Enters segment routing configuration mode.
- Enters adjacency SID configuration mode.
- Enters interface configuration mode.
- Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.
- Configures the Adj-SID index or absolute value for the interface.
- For point-to-point interfaces, you are not required to specify a next-hop. However, if you do specify the next-hop, the Layer 2 Adj-SID will be used only if the specified next-hop matches the neighbor address.
- For LAN interfaces, you must configure the next-hop IPv4 or IPv6 address. If you do not configure the next-hop, the Layer 2 Adj-SID will not be used for LAN interface.
Configure Segment Routing for IS-IS Protocol

<table>
<thead>
<tr>
<th>Step 8</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>end</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 9</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>router isis instance-id</td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>router isis isp</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 10</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>address-family { ipv4</td>
<td>ipv6 } [ unicast ]</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>address-family ipv4 unicast</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 11</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>segment-routing bundle-member-adj-sid</td>
<td>Programs the dynamic Layer 2 Adj-SIDs, and advertises both manual and dynamic Layer 2 Adj-SIDs.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>segment-routing bundle-member-adj-sid</td>
<td></td>
</tr>
</tbody>
</table>

Note
This command is not required to program manual L2 Adj-SID, but is required to program the dynamic Layer 2 Adj-SIDs and to advertise both manual and dynamic Layer 2 Adj-SIDs.

Verify the configuration:

```
Router# show mpls forwarding detail | i "Pop|Outgoing Interface|Physical Interface"
Tue Jun 20 06:53:51.876 PDT
  15001  Pop    SRLB (idx 1)    BE1    10.1.1.4     0
Outgoing Interface: Bundle-Ether1 (ifhandle 0x000000b0)
Physical Interface: GigabitEthernet0/0/0/3 (ifhandle 0x000000b0)
```

```
Router# show running-config segment-routing
Tue Jun 20 07:14:25.815 PDT
segment-routing
  adjacency-sid
  interface GigabitEthernet0/0/0/3
  address-family ipv4 unicast
  l2-adjacency-sid absolute 15001

```

Configuring Bandwidth-Based Local UCMP

Bandwidth-based local Unequal Cost Multipath (UCMP) allows you to enable UCMP functionality locally between Equal Cost Multipath (ECMP) paths based on the bandwidth of the local links.
Bandwidth-based local UCMP is performed for prefixes, segment routing Adjacency SIDs, and Segment Routing label cross-connects installed by IS-IS, and is supported on any physical or virtual interface that has a valid bandwidth.

For example, if the capacity of a bundle interface changes due to the link or line card up/down event, traffic continues to use the affected bundle interface regardless of the available provisioned bundle members. If some bundle members were not available due to the failure, this behavior could cause the traffic to overload the bundle interface. To address the bundle capacity changes, bandwidth-based local UCMP uses the bandwidth of the local links to load balance traffic when bundle capacity changes.

**Before you begin**

### SUMMARY STEPS

1. configure
2. router isis instance-id
3. address-family { ipv4 | ipv6 } [ unicast ]
4. apply-weight ecmp-only bandwidth
5. commit

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
</tbody>
</table>
| **Step 2** | router isis instance-id  
Example:  
RP/0/RSP0/CPU0:router(config)# router isis 1 |
| **Step 3** | address-family { ipv4 | ipv6 } [ unicast ]  
Example:  
The following is an example for ipv4 address family:  
RP/0/RSP0/CPU0:router(config-isis)# address-family ipv4 unicast |
| **Step 4** | apply-weight ecmp-only bandwidth  
Example:  
RP/0/RSP0/CPU0:router(config-isis-af)# apply-weight ecmp-only bandwidth |
| **Step 5** | commit |
IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability

The following sub-TLVs support the advertisement of IPv4 and IPv6 prefix attribute flags and the source router ID of the router that originated a prefix advertisement, as described in RFC 7794.

- Prefix Attribute Flags
- IPv4 and IPv6 Source Router ID

Prefix Attribute Flags

The Prefix Attribute Flag sub-TLV supports the advertisement of attribute flags associated with prefix advertisements. Knowing if an advertised prefix is directly connected to the advertising router helps to determine how labels that are associated with an incoming packet should be processed.

This section describes the behavior of each flag when a prefix advertisement is learned from one level to another.

Note

Prefix attributes are only added when wide metric is used.

Prefix Attribute Flags Sub-TLV Format

```
0 1 2 3 4 5 6 7 ... 
+-------------------------------...
| X | R | N | ... 
+-------------------------------...
```

Prefix Attribute Flags Sub-TLV Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X (External Prefix Flag)</td>
<td>This flag is set if the prefix has been redistributed from another protocol. The value of the flag is preserved when the prefix is propagated to another level.</td>
</tr>
<tr>
<td>R (Re-advertisement Flag)</td>
<td>This flag is set to 1 by the Level 1-2 router when the prefix is propagated between IS-IS levels (from Level 1 to Level 2, or from Level 2 to Level 1). This flag is set to 0 when the prefix is connected locally to an IS-IS-enabled interface (regardless of the level configured on the interface).</td>
</tr>
</tbody>
</table>
For prefixes that are propagated from another level:

1. Copy the N-flag from the prefix attribute sub-TLV, if present in the source level.
2. Copy the N-flag from the prefix-SID sub-TLV, if present in the source level.
3. Otherwise, set to 0.

For connected prefixes:

1. Set to 0 if `isis prefix-attributes n-flag-clear` is configured (see Configuring Prefix Attribute N-flag-clear).
2. Set to 0 if `prefix-SID n-flag-clear` is configured (see Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface).
3. Otherwise, set to 1 when the prefix is a host prefix (/32 for IPV4, /128 for IPv6) that is associated with a loopback address.

**Note:** If the flag is set and the prefix length is not a host prefix, then the flag must be ignored.

### IPv4 and IPv6 Source Router ID

The Source Router ID sub-TLV identifies the source of the prefix advertisement. The IPv4 and IPv6 source router ID is displayed in the output of the `show isis database verbose` command.

The Source Router ID sub-TLV is added when the following conditions are met:

1. The prefix is locally connected.
2. The N-flag is set to 1 (when it's a host prefix and the `n-flag-clear` configuration is not used).
3. The router ID is configured in the corresponding address family.

The source router ID is propagated between levels.

**Table 2: Source Router Sub-TLV Format**

<table>
<thead>
<tr>
<th>IPv4 Source Router ID</th>
<th>Type: 11</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Length: 4</td>
</tr>
<tr>
<td></td>
<td>Value: IPv4 Router ID of the source of the prefix advertisement</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPv6 Source Router ID</th>
<th>Type: 12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Length: 16</td>
</tr>
<tr>
<td></td>
<td>Value: IPv6 Router ID of the source of the prefix advertisement</td>
</tr>
</tbody>
</table>
Configuring Prefix Attribute N-flag-clear

The N-flag is set to 1 when the prefix is a host prefix (/32 for IPv4, /128 for IPv6) that is associated with a loopback address. The advertising router can be configured to not set this flag. This task explains how to clear the N-flag.

SUMMARY STEPS

1. `configure`
2. `interface Loopback instance`
3. `isis prefix-attributes n-flag-clear [Level-1 | Level-2]`
4. `commit`

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 <code>configure</code></td>
<td></td>
</tr>
<tr>
<td>Step 2 <code>interface Loopback instance</code></td>
<td>Specifies the loopback interface and instance.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config)# <code>interface Loopback0</code></td>
<td></td>
</tr>
<tr>
<td>Step 3 `isis prefix-attributes n-flag-clear [Level-1</td>
<td>Level-2]`</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-if)# <code>isis prefix-attributes n-flag-clear</code></td>
<td></td>
</tr>
<tr>
<td>Step 4 <code>commit</code></td>
<td></td>
</tr>
</tbody>
</table>

Verify the prefix attribute configuration:

```
RP/0/RSP0/CPU0:router# `show isis database verbose`
```

```
IS-IS 1 (Level-2) Link State Database
LSPID LSP Seq Num LSP Checksum LSP Holdtime ATT/P/OL
router.00-00 * 0x0000039b 0xfc27 1079 0/0/0
Area Address: 49.0001
NLPID: 0xcc
NLPID: 0x8e
MT: Standard (IPv4 Unicast)
MT: IPv6 Unicast 0/0/0
Hostname: router
IP Address: 10.0.0.1
IPv6 Address: 2001:0db8:1234::0a00:0001
Router Cap: 10.0.0.1, D:0, S:0
Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
SR Algorithm: Algorithm: 0
Algorithm: 1
<...>
Metric: 0 IP-Extended 10.0.0.1/32
```
IS-IS Multi-Domain Prefix SID and Domain Stitching: Example

IS-IS Multi-Domain Prefix SID and Domain Stitching allows you to configure multiple IS-IS instances on the same loopback interface for domain border nodes. You specify a loopback interface and prefix SID under multiple IS-IS instances to make the prefix and prefix SID reachable in different domains.

This example uses the following topology. Node 5 and 9 are border nodes between two IS-IS domains (Domain1 and Domain2). Node 10 is configured as the IOS XR Traffic Controller (XTC) (see Configure Segment Routing Path Computation Element).

Figure 1: Multi-Domain Topology

Configure IS-IS Multi-Domain Prefix SID

Specify a loopback interface and prefix SID under multiple IS-IS instances on each border node:

Example: Border Node 5
router isis Domain1
  interface Loopback0
    address-family ipv4 unicast
      prefix-sid absolute 16005

router isis Domain2
interface Loopback0
  address-family ipv4 unicast
  prefix-sid absolute 16005

Example: Border Node 9
router isis Domain1
  interface Loopback0
    address-family ipv4 unicast
    prefix-sid absolute 16009

router isis Domain2
  interface Loopback0
    address-family ipv4 unicast
    prefix-sid absolute 16009

Border nodes 5 and 9 each run two IS-IS instances (Domain1 and Domain2) and advertise their Loopback0 prefix and prefix SID in both domains.

Nodes in both domains can reach the border nodes by using the same prefix and prefix SID. For example, Node 3 and Node 22 can reach Node 5 using prefix SID 16005.

**Configure Common Router ID**

On each border node, configure a common TE router ID under each IS-IS instance:

Example: Border Node 5
router isis Domain1
  address-family ipv4 unicast
  router-id loopback0

router isis Domain2
  address-family ipv4 unicast
  router-id loopback0

Example: Border Node 9
router isis Domain1
  address-family ipv4 unicast
  router-id loopback0

router isis Domain2
  address-family ipv4 unicast
  router-id loopback0
Distribute IS-IS Link-State Data

Configure BGP Link-state (BGP-LS) on Node 13 and Node 14 to report their local domain to Node 10:

**Example:** Node 13
router isis Domain1
    distribute link-state id

**Example:** Node 14
router isis Domain2
    distribute link-state id

Link-state ID starts from 32. One ID is required per IGP domain. Different domain IDs are essential to identify that the SR-TE TED belongs to a particular IGP domain.

Nodes 13 and 14 each reports its local domain in BGP-LS to Node 10.

Node 10 identifies the border nodes (Nodes 5 and 9) by their common advertised TE router ID, then combines (stitches) the domains on these border nodes for end-to-end path computations.
CHAPTER 5

Configure Segment Routing for OSPF Protocol

Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP) developed by the OSPF working group of the Internet Engineering Task Force (IETF). Designed expressly for IP networks, OSPF supports IP subnetting and tagging of externally derived routing information. OSPF also allows packet authentication and uses IP multicast when sending and receiving packets.

This module provides the configuration information to enable segment routing for OSPF.

For additional information on implementing OSPF on your Cisco ASR 9000 Series Router, see the Implementing OSPF module in the Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide.

Note

- Enabling Segment Routing for OSPF Protocol, on page 31
- Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 33

Enabling Segment Routing for OSPF Protocol

Segment routing on the OSPF control plane supports the following:
- OSPFv2 control plane
- Multi-area
- IPv4 prefix SIDs for host prefixes on loopback interfaces
- Adjacency SIDs for adjacencies
- MPLS penultimate hop popping (PHP) and explicit-null signaling

This section describes how to enable segment routing MPLS and MPLS forwarding in OSPF. Segment routing can be configured at the instance, area, or interface level.

Before you begin

Your network must support the MPLS Cisco IOS XR software feature before you enable segment routing for OSPF on your router.
You must enter the commands in the following task list on every OSPF router in the traffic-engineered portion of your network.

### SUMMARY STEPS

1. `configure`
2. `router ospf process-name`
3. `segment-routing mpls`
4. `area 0`
5. `mpls traffic-eng area`
6. `mpls traffic-eng router-id interface`
7. `segment-routing mpls`
8. `exit`
9. `mpls traffic-eng`
10. `commit`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enables OSPF routing for the specified routing process and places the router in router configuration mode.</td>
</tr>
<tr>
<td><strong>Step 2</strong> router ospf process-name</td>
<td>Enables OSPF routing for the specified routing process and places the router in router configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# <code>router ospf 1</code></td>
</tr>
<tr>
<td><strong>Step 3</strong> segment-routing mpls</td>
<td>Enables segment routing using the MPLS data plane on the routing process and all areas and interfaces in the routing process. Enables segment routing forwarding on all interfaces in the routing process and installs the SIDs received by OSPF in the forwarding table.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# <code>segment-routing mpls</code></td>
</tr>
<tr>
<td><strong>Step 4</strong> area 0</td>
<td>Enters area configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# <code>area</code></td>
</tr>
<tr>
<td><strong>Step 5</strong> mpls traffic-eng area</td>
<td>Enables IGP traffic engineering functionality.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar)# <code>mpls traffic-eng area</code></td>
</tr>
<tr>
<td><strong>Step 6</strong> mpls traffic-eng router-id interface</td>
<td>Sets the traffic engineering loopback interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
</tbody>
</table>
Configure Segment Routing for OSPF Protocol

Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface

A prefix SID is associated with an IP prefix. The prefix SID is manually configured from the segment routing global block (SRGB) range of labels. The prefix segment steers the traffic along the shortest path to its destination. A node SID is a special type of prefix SID that identifies a specific node. It is configured under the loopback interface with the loopback address of the node as the prefix.

The prefix SID is globally unique within the segment routing domain.

This task describes how to configure prefix segment identifier (SID) index or absolute value on the OSPF-enabled Loopback interface.

**Before you begin**

Ensure that segment routing is enabled on an instance, area, or interface.

**SUMMARY STEPS**

1. configure
2. router ospf  process-name
3. area  value

---

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar)# mpls traffic-eng router-id Loopback0</td>
<td>(Optional) Enables segment routing using the MPLS data plane on the area and all interfaces in the area. Enables segment routing forwarding on all interfaces in the area and installs the SIDs received by OSPF in the forwarding table.</td>
</tr>
<tr>
<td>Step 7 segment-routing mpls</td>
<td>Example: RP/0/RSP0/CPU0:router(config-ospf-ar)# segment-routing mpls</td>
</tr>
<tr>
<td>Step 8 exit</td>
<td>Example: RP/0/RSP0/CPU0:router(config-ospf-ar)# exit</td>
</tr>
<tr>
<td>Step 9 mpls traffic-eng</td>
<td>Example: RP/0/RSP0/CPU0:router(config)# mpls traffic-eng</td>
</tr>
<tr>
<td>Step 10 commit</td>
<td>Enables traffic engineering functionality on the node. The node advertises the traffic engineering link attributes in IGP which populates the traffic engineering database (TED) on the head-end. The SR-TE head-end requires the TED to calculate and validate the path of the SR-TE policy.</td>
</tr>
</tbody>
</table>

**What to do next**

Configure the prefix SID.
### Configure Segment Routing for OSPF Protocol

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
<td>Enables OSPF routing for the specified routing process, and places the router in router configuration mode.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><code>router ospf  process-name</code>&lt;br&gt;Example:&lt;br&gt;R/P0/RSP0/CPU0:router(config)# router ospf 1</td>
<td>Enables OSPF routing for the specified routing process, and places the router in router configuration mode.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><code>area  value</code>&lt;br&gt;Example:&lt;br&gt;R/P0/RSP0/CPU0:router(config-ospf)# area 0</td>
<td>Enters area configuration mode.</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>interface Loopback  interface-instance&lt;br&gt;Example:&lt;br&gt;R/P0/RSP0/CPU0:router(config-ospf-ar)# interface Loopback0 passive</td>
<td>Specifies the loopback interface and instance.</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td>prefix-sid  `{index  SID-index</td>
<td>absolute  SID-value } [n-flag-clear ] [ explicit-null ]`&lt;br&gt;Example:&lt;br&gt;R/P0/RSP0/CPU0:router(config-ospf-ar)# prefix-sid index 1001&lt;br&gt;R/P0/RSP0/CPU0:router(config-ospf-ar)# prefix-sid absolute 17001</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td>commit</td>
<td>Verify the prefix-SID configuration:</td>
</tr>
</tbody>
</table>

```bash
R/P0/RSP0/CPU0:router# show ospf database opaque-area 7.0.0.1 self-originate
OSPF Router with ID (10.0.0.1) (Process ID 1)
Type-10 Opaque Link Area Link States (Area 0)
<...>
Extended Prefix TLV: Length: 20
```
Route-type: 1
AF : 0
Flags : 0x40
Prefix : 10.0.0.1/32

SID sub-TLV: Length: 8
Flags : 0x0
MTID : 0
Algo : 0
Sid Index : 1001

What to do next

Configure SR-TE Policies
Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface
CHAPTER 6

Configure Segment Routing for BGP

Border Gateway Protocol (BGP) is an Exterior Gateway Protocol (EGP) that allows you to create loop-free inter-domain routing between autonomous systems. An autonomous system is a set of routers under a single technical administration. Routers in an autonomous system can use multiple Interior Gateway Protocols (IGPs) to exchange routing information inside the autonomous system and an EGP to route packets outside the autonomous system.

This module provides the configuration information used to enable segment routing for BGP.

---

**Note**

For additional information on implementing BGP on your Cisco ASR 9000 Series Router, see the Implementing BGP module in the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*.

- Segment Routing for BGP, on page 37
- Configure BGP Prefix Segment Identifiers, on page 38
- Configure Segment Routing Egress Peer Engineering, on page 39
- Configure BGP Link-State, on page 40
- Example: Configuring SR-EPE and BGP-LS, on page 41
- Configure BGP Proxy Prefix SID, on page 43

---

**Segment Routing for BGP**

In a traditional BGP-based data center (DC) fabric, packets are forwarded hop-by-hop to each node in the autonomous system. Traffic is directed only along the external BGP (eBGP) multipath ECMP. No traffic engineering is possible.

In an MPLS-based DC fabric, the eBGP sessions between the nodes exchange BGP labeled unicast (BGP-LU) network layer reachability information (NLRI). An MPLS-based DC fabric allows any leaf (top-of-rack or border router) in the fabric to communicate with any other leaf using a single label, which results in higher packet forwarding performance and lower encapsulation overhead than traditional BGP-based DC fabric.

However, since each label value might be different for each hop, an MPLS-based DC fabric is more difficult to troubleshoot and more complex to configure.

BGP has been extended to carry segment routing prefix-SID index. BGP-LU helps each node learn BGP prefix SIDs of other leaf nodes and can use ECMP between source and destination. Segment routing for BGP simplifies the configuration, operation, and troubleshooting of the fabric. With segment routing for BGP, you can enable traffic steering capabilities in the data center using a BGP prefix SID.
Configure BGP Prefix Segment Identifiers

Segments associated with a BGP prefix are known as BGP prefix SIDs. The BGP prefix SID is global within a segment routing or BGP domain. It identifies an instruction to forward the packet over the ECMP-aware best-path computed by BGP to the related prefix. The BGP prefix SID is manually configured from the segment routing global block (SRGB) range of labels.

Each BGP speaker must be configured with an SRGB using the `segment-routing global-block` command. See the About the Segment Routing Global Block section for information about the SRGB.

**Note**

Because the values assigned from the range have domain-wide significance, we recommend that all routers within the domain be configured with the same range of values.

To assign a BGP prefix SID, first create a routing policy using the `set label-index index` attribute, then associate the index to the node.

**Note**

A routing policy with the `set label-index` attribute can be attached to a network configuration or redistribute configuration. Other routing policy language (RPL) configurations are possible. For more information on routing policies, refer to the "Implementing Routing Policy" chapter in the Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide.

**Example**

The following example shows how to configure the SRGB, create a BGP route policy using a $SID parameter and `set label-index` attribute, and then associate the prefix-SID index to the node.

```
RP/0/RSP0/CPU0:router(config)# segment-routing global-block 16000 23999
RP/0/RSP0/CPU0:router(config)# route-policy SID($SID)
RP/0/RSP0/CPU0:router(config-rpl)# set label-index $SID
RP/0/RSP0/CPU0:router(config-rpl)# end policy
RP/0/RSP0/CPU0:router(config)# router bgp 1
RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 1.1.1.1
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)# network 1.1.1.3/32 route-policy SID(3)
RP/0/RSP0/CPU0:router(config-bgp-af)# allocate-label all
RP/0/RSP0/CPU0:router(config-bgp-af)# commit
RP/0/RSP0/CPU0:router(config-bgp-af)# end

RP/0/RSP0/CPU0:router# show bgp 1.1.1.3/32
BGP routing table entry for 1.1.1.3/32
Versions:
  Process  bRIB/RIB  SendTblVer
  Speaker  74  74
Local Label:16003
Last Modified: Sep 29 19:52:18.155 for 00:07:22
Paths: (1 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
```
Configure Segment Routing Egress Peer Engineering

Segment routing egress peer engineering (EPE) uses a controller to instruct an ingress provider edge, or a content source (node) within the segment routing domain, to use a specific egress provider edge (node) and a specific external interface to reach a destination. BGP peer SIDs are used to express source-routed inter-domain paths.

The controller learns the BGP peer SIDs and the external topology of the egress border router through BGP-LS EPE routes. The controller can program an ingress node to steer traffic to a destination through the egress node and peer node using BGP labeled unicast (BGP-LU).

EPE functionality is only required at the EPE egress border router and the EPE controller.

This task explains how to configure segment routing EPE on the EPE egress node.

SUMMARY STEPS

1. `router bgp as-number`
2. `neighbor ip-address`
3. `remote-as as-number`
4. `egress-engineering`

DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><code>router bgp as-number</code></td>
<td>Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>router bgp 1</code></td>
<td></td>
</tr>
<tr>
<td>Step 2</td>
<td><code>neighbor ip-address</code></td>
<td>Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>neighbor 192.168.1.3</code></td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td><code>remote-as as-number</code></td>
<td>Creates a neighbor and assigns a remote autonomous system number to it.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Configure BGP Link-State

BGP Link-State (LS) is an Address Family Identifier (AFI) and Sub-address Family Identifier (SAFI) defined to carry interior gateway protocol (IGP) link-state database through BGP. BGP LS delivers network topology information to topology servers and Application Layer Traffic Optimization (ALTO) servers. BGP LS allows policy-based control to aggregation, information-hiding, and abstraction. BGP LS supports IS-IS and OSPFv2.

IGPs do not use BGP LS data from remote peers. BGP does not download the received BGP LS data to any other component on the router.

For segment routing, the following attributes have been added to BGP LS:

- Node—Segment routing capability (including SRGB range) and algorithm
- Link—Adjacency SID and LAN adjacency SID
- Prefix—Prefix SID and segment routing mapping server (SRMS) prefix range

The following example shows how to exchange link-state information with a BGP neighbor:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# router bgp 1
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.0.0.2
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 1
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family link-state link-state
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# exit
```

IGP Extensions

A given BGP node may have connections to multiple, independent routing domains; IGP link-state distribution into BGP has been added for both OSPF and ISIS protocols to enable that node to pass this information, in a similar fashion, on to applications that desire to build paths spanning or including these multiple domains.

To distribute ISIS link-state data using BGP LS, use the distribute bgp-ls command in router configuration mode.

```
RP/0/RSP0/CPU0:router# configure
```
**Example: Configuring SR-EPE and BGP-LS**

In the following figure, segment routing is enabled on autonomous system AS1 with ingress node A and egress nodes B and C. In this example, we configure EPE on egress node C.

*Figure 2: Topology*

**Step 1** Configure node C with EPE for eBGP peers D and E.

**Example:**

```sh
RP/0/RSP0/CPU0:router_C(config)# router bgp 1
RP/0/RSP0/CPU0:router_C(config-bgp)# neighbor 192.168.1.3
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# remote-as 3
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# description to E
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# egress-engineering
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_in in
RP/0/RSP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_out out
RP/0/RSP0/CPU0:router_C(config-bgp-nbr-af)# exit
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# exit
RP/0/RSP0/CPU0:router_C(config-bgp)# neighbor 192.168.1.2
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# remote-as 2
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# description to D
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# egress-engineering
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_in in
RP/0/RSP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_out out
RP/0/RSP0/CPU0:router_C(config-bgp-nbr-af)# exit
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# exit
```

**Step 2** Configure node C to advertise peer node SIDs to the controller using BGP-LS.

```sh
RP/0/RSP0/CPU0:router_C(config)# router ospf 100
RP/0/RSP0/CPU0:router_C(config-ospf)# distribute bgp-ls instance-id 32 throttle 10
```
Example: Configuring SR-EPE and BGP-LS

Example:

RP/0/RSP0/CPU0:router_C(config-bgp)# neighbor 172.29.50.71
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# remote-as 1
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# description to EPE_controller
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# address-family link-state link-state
RP/0/RSP0/CPU0:router_C(config-bgp-nbr)# exit
RP/0/RSP0/CPU0:router_C(config-bgp)# exit

Step 3 Commit the configuration.

Example:

RP/0/RSP0/CPU0:router_C(config)# commit

Step 4 Verify the configuration.

Example:

RP/0/RSP0/CPU0:router_C# show bgp egress-engineering

Egress Engineering Peer Set: 192.168.1.2/32 (10b87210)
   Nexthop: 192.168.1.2
   Version: 2, rn_version: 2
   Flags: 0x00000002
   Local ASN: 1
   Remote ASN: 2
   Local RID: 1.1.1.3
   Remote RID: 1.1.1.4
   First Hop: 192.168.1.2
   NHID: 3
   Label: **24002**, Refcount: 3
   rpc_set: 10b9d408

Egress Engineering Peer Set: 192.168.1.3/32 (10be61d4)
   Nexthop: 192.168.1.3
   Version: 3, rn_version: 3
   Flags: 0x00000002
   Local ASN: 1
   Remote ASN: 3
   Local RID: 1.1.1.3
   Remote RID: 1.1.1.5
   First Hop: 192.168.1.3
   NHID: 4
   Label: **24003**, Refcount: 3
   rpc_set: 10be6250

The output shows that node C has allocated peer SIDs for each eBGP peer.

Example:

RP/0/RSP0/CPU0:router_C# show mpls forwarding labels 24002 24003

<table>
<thead>
<tr>
<th>Label</th>
<th>Prefix</th>
<th>Outgoing</th>
<th>Next Hop</th>
<th>Switched</th>
</tr>
</thead>
<tbody>
<tr>
<td>24002</td>
<td>Unlabelled</td>
<td>Te0/3/0/0</td>
<td>192.168.1.2</td>
<td>0</td>
</tr>
<tr>
<td>24003</td>
<td>Unlabelled</td>
<td>Te0/1/0/0</td>
<td>192.168.1.3</td>
<td>0</td>
</tr>
</tbody>
</table>
Configure BGP Proxy Prefix SID

To support segment routing, Border Gateway Protocol (BGP) requires the ability to advertise a segment identifier (SID) for a BGP prefix. A BGP-Prefix-SID is the segment identifier of the BGP prefix segment in a segment routing network. BGP prefix SID attribute is a BGP extension to signal BGP prefix-SIDs. However, there may be routers which do not support BGP extension for segment routing. Hence, those routers also do not support BGP prefix SID attribute and an alternate approach is required.

BGP proxy prefix SID feature allows you to attach BGP prefix SID attributes for remote prefixes learnt from BGP labeled unicast (LU) neighbours which are not SR-capable and propagate them as SR prefixes. This allows an LSP towards non SR endpoints to use segment routing global block in a SR domain. Since BGP proxy prefix SID uses global label values it minimizes the use of limited resources such as ECMP-FEC and provides more scalability for the networks.

BGP proxy prefix SID feature is implemented using the segment routing mapping server (SRMS). SRMS allows the user to configure SID mapping entries to specify the prefix-SIDs for the prefixes. The mapping server advertises the local SID-mapping policy to the mapping clients. BGP acts as a client of the SRMS and uses the mapping policy to calculate the prefix-SIDs.

Configuration Example:

This example shows how to configure the BGP proxy prefix SID feature for the segment routing mapping server.

```
RP/0/RSP0/CPU0:router(config)# segment-routing
RP/0/RSP0/CPU0:router(config-sr)# mapping-server
RP/0/RSP0/CPU0:router(config-sr-ms)# prefix-sid-map
RP/0/RSP0/CPU0:router(config-sr-ms-map)# address-family ipv4
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)# 1.1.1.1/32 10 range 200
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)# 192.168.64.1/32 400 range 300
```

This example shows how to configure the BGP proxy prefix SID feature for the segment-routing mapping client.

```
RP/0/RSP0/CPU0:router(config)# router bgp 1
RP/0/RSP0/CPU0:router(config-bgp)# address-family ip4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)# segment-routing prefix-sid-map
```

Verification

These examples show how to verify the BGP proxy prefix SID feature.

```
RP/0/RSP0/CPU0:router# show segment-routing mapping-server prefix-sid-map ipv4
detail
Prefix 1.1.1.1/32
  SID Index: 10
  Range: 200
  Last Prefix: 1.1.1.200/32
  Last SID Index: 209
```
Number of mapping entries: 1

**RP/0/RSP0/CPU0:router# show bgp ipv4 labeled-unicast 192.168.64.1/32**

BGP routing table entry for 192.168.64.1/32

 Versions:
     Process bRIB/RIB SendTblVer
     Speaker           117      117

 Local Label: 16400
 Last Modified: Oct 25 01:02:28.562 for 00:11:45

 Paths: (2 available, best #1)
 Advertised to peers (in unique update groups):
     201.1.1.1
     200.0.101.1 from 200.0.101.1 (20.0.101.1)
 Origin IGP, localpref 100, valid, internal, best, group-best, multipath, labeled-unicast

 Received Path ID 0, Local Path ID 0, version 117
 Prefix SID Attribute Size: 7
 Label Index: 1

**RP/0/RSP0/CPU0:router# show route ipv4 unicast 192.68.64.1/32 detail**

Routing entry for 192.168.64.1/32

 Known via "bgp 65000", distance 200, metric 0, [ei]-bgp, labeled SR, type internal
 Installed Oct 25 01:02:28.583 for 00:20:09

 Routing Descriptor Blocks
     200.0.101.1, from 200.0.101.1, BGP multi path
     Route metric is 0
     Label: 0x3d (61)
     Tunnel ID: None
     Binding Label: None
     Extended communities count: 0
     NHID:0x0 (Ref:0)
     Route version is 0x6 (6)
     Local Label: 0x3e81 (16400)
     IP Precedence: Not Set
     QoS Group ID: Not Set
     Flow-tag: Not Set
     Fwd-class: Not Set
     Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type RIB_SVD_TYPE_LOCAL
     Download Priority 4, Download Version 242
     No advertising protos.

**RP/0/RSP0/CPU0:router# show cef ipv4 192.168.64.1/32 detail**

192.168.64.1/32, version 476, labeled SR, drop adjacency, internal 0x5000001 0x80 (ptr
     0x71ae7e78) reference count 3, flags 0xa7, source rib (7), 0 backups
     [2 type 5 flags 0x88401 (0x722eb450) ext 0x80 (0x0)]
     LW-LDI[type=5, refc=3, ptr=0x71c1590, sh-ldi=0x722eb450]
     gateway array update time 3 Oct 31 23:48:7.733
     Prefix Len 32, traffic index 0, precedence n/a, priority 4
     Extensions: context-label:16400
     gateway array (0x71ae7e78) reference count 3, flags 0xa7, source rib (7), 0 backups
     [2 type 5 flags 0x88401 (0x722eb450) ext 0x80 (0x0)]
     LW-LDI[type=5, refc=3, ptr=0x71c1590, sh-ldi=0x722eb450]
     gateway array update time 3 Oct 31 23:48:7.733
     LDI Update time Oct 31 23:48:7.733
     path-idx 0 NHID 0x0 (0x7129a294 [0x71ae7e78])
     recursion-via=/32
     unresolved
     local label 16400
labels imposed {ExpNullv6}

RP/0/RSP0/CPU0:router# show bgp labels
BGP router identifier 2.1.1.1, local AS number 65000
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: Ox00000000  RD version: 245
BGP main routing table version 245
BGP NSR Initial initsync version 16 (Reached)
BGP NSR/ISSU Sync-Group versions 245/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
    i = internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i = IGP, e = EGP, ? = incomplete

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Hop</th>
<th>Rcvd Label</th>
<th>Local Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.1.1/32</td>
<td>1.1.1.1</td>
<td>3</td>
<td>16010</td>
</tr>
<tr>
<td>2.1.1.1/32</td>
<td>0.0.0.0</td>
<td>nolabel</td>
<td>3</td>
</tr>
<tr>
<td>192.68.64.1/32</td>
<td>20.0.101.1</td>
<td>2</td>
<td>16400</td>
</tr>
<tr>
<td>192.68.64.2/32</td>
<td>20.0.101.1</td>
<td>2</td>
<td>16401</td>
</tr>
</tbody>
</table>
CHAPTER 7

Configure SR-TE Policies

This module provides information about segment routing for traffic engineering (SR-TE) policies, how to configure SR-TE policies, and how to steer traffic into an SR-TE policy.

- Configure SR-TE Policies, on page 47
- BGP SR-TE, on page 57
- Using Binding Segments, on page 61

Configure SR-TE Policies

Segment routing for traffic engineering (SR-TE) uses a “policy” to steer traffic through the network. An SR-TE policy path is expressed as a list of segments that specifies the path, called a segment ID (SID) list. Each segment is an end-to-end path from the source to the destination, and instructs the routers in the network to follow the specified path instead of following the shortest path calculated by the IGP. If a packet is steered into an SR-TE policy, the SID list is pushed on the packet by the head-end. The rest of the network executes the instructions embedded in the SID list.

An SR-TE policy is identified as an ordered list (head-end, color, end-point):

- Head-end – Where the SR-TE policy is instantiated
- Color – A numerical value that distinguishes between two or more policies to the same node pairs (Head-end – End point)
- End-point – The destination of the SR-TE policy

Every SR-TE policy has a color value. Every policy between the same node pairs requires a unique color value.

An SR-TE policy uses one or more candidate paths. A candidate path is a single segment list (SID-list) or a set of weighted SID-lists (for weighted equal cost multi-path [WECMP]). A candidate path is either dynamic or explicit.

A dynamic path is based on an optimization objective and a set of constraints. The head-end computes a solution, resulting in a SID-list or a set of SID-lists. When the topology changes, a new path is computed. If the head-end does not have enough information about the topology, the head-end might delegate the computation to a path computation engine (PCE). For information on configuring the XTC as a PCE, see the Configure IOS XR Traffic Controller (XTC) chapter.

An explicit path is a specified SID-list or set of SID-lists.
An SR-TE policy initiates a single (selected) path in RIB/FIB. This is the preferred valid candidate path.

A candidate path has the following characteristics:

- It has a preference – If two policies have same {color, endpoint} but different preferences, the policy with the highest preference is selected.
- It is associated with a single binding SID (BSID) – A BSID conflict occurs when there are different SR policies with the same BSID. In this case, the policy that is installed first gets the BSID and is selected.
- It is valid if it is usable.

A path is selected when the path is valid and its preference is the best among all candidate paths for that policy.

**Note**

The protocol of the source is not relevant in the path selection logic.

**Configuration Example**

To configure a local SR-TE policy, you must complete the following configurations:

1. Create the segment lists.
2. Create the policy.

**Configure Local SR-TE Policy**

```plaintext
/* Enter the global configuration mode and create the SR-TE segment lists */
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list name Plist-1
Router(config-sr-te-sl)# index 1 mpls label 400102
Router(config-sr-te-sl)# index 2 mpls label 400106
Router(config-sr-te-sl)# exit

Router(config-sr-te)# segment-list name Plist-2
Router(config-sr-te-sl)# index 1 mpls label 400222
Router(config-sr-te-sl)# index 2 mpls label 400106
Router(config-sr-te-sl)# exit

/* Create the SR-TE policy */
Router(config-sr-te)# policy P1
Router(config-sr-te-policy)# binding-sid mpls 15001
Router(config-sr-te-policy)# color 1 end-point ipv4 6.6.6.6
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 10
Router(config-sr-te-pp-index)# explicit segment-list Plist-1
Router(config-sr-te-pp-info)# weight 2
Router(config-sr-te-pp-info)# exit

Router(config-sr-te-pp-index)# explicit segment-list Plist-2
Router(config-sr-te-pp-info)# weight 2
Router(config-sr-te-pp-info)# commit
Router(config-sr-te-pp-info)# end
Router(config)#
```
**Running Configuration**

Router# show running-configuration
segment-routing
  traffic-eng
    segment-list name Plist-1
      index 1 mpls label 400102
      index 2 mpls label 400106
    !
    segment-list name Plist-2
      index 1 mpls label 400222
      index 2 mpls label 400106
    !
    policy P1
      binding-sid mpls 15001
      color 1 end-point ipv4 6.6.6.6
      candidate-paths
        preference 10
        explicit segment-list Plist-1
          weight 2
          !
        explicit segment-list Plist-2
          weight 2
          !
          !
          !
          !

**Verification**

Router# show segment-routing traffic-eng policy name srte_c_1_ep_6.6.6.6
Sat Jul 8 12:25:34.114 UTC
SR-TE policy database
---------------------
Name: P1 (Color: 1, End-point: 6.6.6.6)
Status:
  Admin: up Operational: up for 00:06:21 (since Jul 8 12:19:13.198)
Candidate-paths:
  Preference 10:
    Explicit: segment-list Plist-1 (active)
      Weight: 2
      400102 [Prefix-SID, 2.1.1.1]
      400106
    Explicit: segment-list Plist-2 (active)
      Weight: 2
      400222 [Prefix-SID, 22.11.1.1]
      400106
Attributes:
  Binding SID: 15001
  Allocation mode: explicit
  State: programmed
  Policy selected: yes
  Forward Class: 0
Autoroute Include

You can configure SR-TE policies with Autoroute Include to steer specific IGP (IS-IS, OSPF) prefixes over non-shortest paths and to divert the traffic for those prefixes on to the SR-TE policy. Autoroute Include applies Autoroute Announce functionality to the specified destinations or prefixes.

The Autoroute SR-TE policy adds the prefixes into the IGP, which determines if the prefixes on the endpoint or downstream of the endpoint are eligible to use the SR-TE policy. If a prefix is eligible, then the IGP checks if the prefix is listed in the Autoroute Include configuration. If the prefix is included, then the IGP downloads the prefix route with the SR-TE policy as the outgoing path.

Autoroute Include supports three metric types:

- Default (no metric): The path over the SR-TE policy inherits the shortest path metric.
- Absolute metric: The shortest path metric to the policy endpoint is replaced with the configured absolute metric. The metric to any prefix that is Autoroute Included is modified to the absolute metric.
- Relative metric: The shortest path metric to the policy endpoint is modified with the relative value configured (plus or minus).

Note

To prevent load-balancing over IGP paths, you can specify a metric that is lower than the value that IGP takes into account for autorouted destinations (for example, `autoroute metric relative -1`).

Configuration Example

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# policy P1
Router(config-sr-te-policy)# color 20 end ipv4 1.1.1.2
Router(config-sr-te-policy)# autoroute include ipv4 1.1.1.2/32
Router(config-sr-te-policy)# autoroute include ipv4 1.1.1.23/32
Router(config-sr-te-policy)# autoroute metric constant 1
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-pp-index)# explicit segment-list Plist-1
```

Color-Only Steering

Color-only steering is a traffic steering mechanism where a policy is created with given color, regardless of the endpoint.

You can create an SR-TE policy for a specific color that uses a NULL end-point (0.0.0.0 for IPv4 NULL, and ::0 for IPv6 NULL end-point). This means that you can have a single policy that can steer traffic that is based on that color and a NULL endpoint for routes with a particular color extended community, but different destinations (next-hop).
Every SR-TE policy with a NULL end-point must have an explicit path-option. The policy cannot have a dynamic path-option (where the path is computed by the head-end or PCE) since there is no destination for the policy.

You can also specify a color-only (CO) flag in the color extended community for overlay routes. The CO flag allows the selection of an SR-policy with a matching color, regardless of endpoint Sub-address Family Identifier (SAFI) (IPv4 or IPv6). See Setting CO Flag, on page 60.

**Configure Color-Only Steering**

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# policy P1
Router(config-sr-te-policy)# color 1 end-point ipv4 0.0.0.0

Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# policy P2
Router(config-sr-te-policy)# color 2 end-point ipv6 ::0
```

```
Router# show running-configuration
segment-routing
  traffic-eng
    policy P1
      color 1 end-point ipv4 0.0.0.0
    !
    policy P2
      color 2 end-point ipv6 ::
    !
    !
end
```

**Address-Family Agnostic Steering**

Address-family agnostic steering uses an SR-TE policy to steer both labeled and unlabeled IPv4 and IPv6 traffic. This feature requires support of IPv6 encapsulation (IPv6 caps) over IPv4 endpoint policy.

IPv6 caps for IPv4 NULL end-point is enabled automatically when the policy is created in XR Traffic Controller (XTC). The binding SID (BSID) state notification for each policy contains an "ipv6_caps" flag that notifies XTC clients of the status of IPv6 caps (enabled or disabled).

An SR-TE policy with a given color and IPv4 NULL end-point could have more than one candidate path. If any of the candidate paths has IPv6 caps enabled, then all of the remaining candidate paths need IPv6 caps enabled. If IPv6 caps is not enabled on all candidate paths of same color and end-point, traffic drops can occur.

You can disable IPv6 caps for a particular color and IPv4 NULL end-point using the **ipv6 disable** command on the local policy. This command disables IPv6 caps on all candidate paths that share the same color and IPv4 NULL end-point.
Disable IPv6 Encapsulation

Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# policy P1
Router(config-sr-te-policy)# color 1 end-point ipv4 0.0.0.0
Router(config-sr-te-policy)# ipv6 disable

Flexible Name-based Policy Constraints

SR-TE Flexible Name-based Policy Constraints provides a simplified and more flexible means of configuring link attributes and path affinities to compute paths for SR-TE policies.

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).

SR-TE Flexible Name-based Policy 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 CLI. Furthermore, you can define constraints using include-any, include-all, and exclude-any arguments, where each statement can contain up to 10 colors.

Note

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

Configuring Flexible Name-Based Policy Constraints

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

1. Assign Color Names to Numeric Values
2. Associate Affinity-Names with SR-TE Links
3. Associate Affinity Constraints for SR-TE Policies

/* Enter the global configuration mode and assign color names to numeric values
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# affinity-map
Router(config-sr-te-affinity-map)# blue bit-position 0
Router(config-sr-te-affinity-map)# green bit-position 1
Router(config-sr-te-affinity-map)# red bit-position 2
Router(config-sr-te-affinity-map)# exit

/* Associate affinity-names with SR-TE links
Router(config-sr-te)# interface Gi0/0/0/0
Router(config-sr-te-if)# affinity
Router(config-sr-te-if-affinity)# blue
Router(config-sr-te-if-affinity)# exit
Router(config-sr-te-if)# exit
Router(config-sr-te)# interface Gi0/0/0/1
Router(config-sr-te-if)# affinity
Router(config-sr-te-if-affinity)# blue
Router(config-sr-te-if-affinity)# green
Router(config-sr-te-if-affinity)# exit
Router(config-sr-te-if)# exit
Router(config-sr-te)#

/* Associate affinity constraints for SR-TE policies
Router(config-sr-te)# segment-list name SIDLIST1
Router(config-sr-te-sl)# index 10 address ipv4 1.1.1.2
Router(config-sr-te-sl)# index 20 address ipv4 2.2.2.23
Router(config-sr-te-sl)# index 30 address ipv4 1.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list name SIDLIST2
Router(config-sr-te-sl)# index 10 address ipv4 1.1.1.2
Router(config-sr-te-sl)# index 30 address ipv4 1.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list name SIDLIST3
Router(config-sr-te-sl)# index 10 address ipv4 1.1.1.5
Router(config-sr-te-sl)# index 30 address ipv4 1.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# color 20 end-point ipv4 1.1.1.4
Router(config-sr-te-policy)# binding-sid mpls 1000
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 200
Router(config-sr-te-policy-path-pref)# constraints affinity exclude-any red
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST1
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST2
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path)# explicit segment-list SIDLIST3

Running Configuration

Router# show running-configuration
segment-routing
  traffic-eng
    interface GigabitEthernet0/0/0/0
      affinity
        blue
      
interface GigabitEthernet0/0/0/1
  affinity
    blue
    green

segment-list name SIDLIST1
Affinity Support for Anycast SIDs

For information about configuring anycast SIDs, see Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 16 or Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 33.

Routers that are configured with the same anycast SID, on the same Loopback address and with the same SRGB, advertise the same prefix SID (anycast).

The shortest path with the lowest IGP metric is then verified against the affinity constraints. If multiple nodes have the same shortest-path metric, all their paths are validated against the affinity constraints. A path that is not the shortest path is not validated against the affinity constraints.
Affinity Support for Anycast SIDs: Examples

In the following examples, nodes 3 and 5 advertise the same anycast prefix (1.1.1.8) and assign the same prefix SID (16100).

Node 1 uses the following SR-TE policy:

```
segment-routing
traffic-eng
policy POLICY1
  color 20 end-point ipv4 1.1.1.4
  binding-sid mpls 1000
  candidate-paths
  preference 100
  explicit segment-list SIDLIST1
  constraints
  affinity
  exclude-any
  red
  segment-list name SIDLIST1
  index 10 address ipv4 100.100.100.100
  index 20 address ipv4 4.4.4.4
```

Valid Paths: Example

In this example, the shortest path to both node 3 and node 5 has an equal accumulative IGP metric of 20. Both paths are validated against affinity constraints.
Invalid Path Based on Affinity Constraint: Example

In this example, parallel link (23) is marked as red, so the path to anycast node 3 is invalidated.

SR-TE policy database
----------------------
Name: POLICY1 (Color: 2, End-point: 198.51.100.6)
Status:
  Admin: up Operational: up for 00:03:52 (since Jan 24 01:52:14.215)
Candidate-paths:
  Preference 100:
  Constraints:
    Affinity:
      exclude-any: red
    Explicit: segment-list SIDLIST1 (inactive)
  Inactive Reason: Link [2.2.21.23,2.2.21.32] failed to satisfy affinity exclude-any constraint=0x00000008, link attributes=0x0000000a

Invalid Path Based on IGP Metric: Example

In this example, the shortest path to node 5 has an accumulative IGP metric of 20, and the shortest path to node 3 has an accumulative IGP metric of 30. Only the shortest path to node 5 is validated against affinity constraints.
Even though parallel link (23) is marked with red, it is still considered valid since anycast traffic flows only on the path to node 5.

**BGP SR-TE**

SR-TE can be used by data center (DC) operators to provide different levels of Service Level Assurance (SLA). Setting up SR-TE paths using BGP (BGP SR-TE) simplifies DC network operation without introducing a new protocol for this purpose.

For more information on routing policies and routing policy language (RPL), refer to the "Implementing Routing Policy" chapter in the Routing Configuration Guide for Cisco ASR 9000 Series Aggregation Services Router.

**Explicit BGP SR-TE**

Explicit BGP SR-TE uses an SR-TE policy (identified by a unique color ID) that contains a list of explicit paths with SIDs that correspond to each explicit path. A BGP speaker signals an explicit SR-TE policy to a remote peer, which triggers the setup of an TE tunnel with specific characteristics and explicit paths. On the receiver side, a TE tunnel that corresponds to the explicit path is setup by BGP. The packets for the destination mentioned in the BGP update follow the explicit path described by the policy. Each policy can include multiple explicit paths, and TE will create a tunnel for each path.

**Configure Explicit BGP SR-TE**

Perform this task to configure explicit BGP SR-TE:

**Before you begin**

The following configuration must be applied on the head-end router:

```
Router(config)# ipv4 unnumbered mpls traffic-eng Loopback0
Router(config)# mpls traffic-eng
```
**SUMMARY STEPS**

1. configure
2. extcommunity-set opaque name
3. name
4. end-set
5. route-policy route-policy-name
6. end-policy
7. router bgp as-number
8. bgp router-id ip-address
9. address-family {ipv4 | ipv6} sr-policy
10. exit
11. neighbor ip-address
12. remote-as as-number
13. address-family {ipv4 | ipv6} unicast
14. route-policy route-policy-name {in | out}
15. send-extended-community-ebgp

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 extcommunity-set opaque name</td>
<td>Defines the color extended community-set.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config) # extcommunity-set opaque color1</td>
<td></td>
</tr>
<tr>
<td>Step 3 name</td>
<td>Defines the color extended community-set.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-ext)# 1</td>
<td></td>
</tr>
<tr>
<td>Step 4 end-set</td>
<td>Ends the definition of the extended community-set.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-ext)# end-set</td>
<td></td>
</tr>
<tr>
<td>Step 5 route-policy route-policy-name</td>
<td>Creates a route policy and enters route policy configuration mode, where you can define the route policy to mark the prefixes with the color extended community value.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config) # route-policy color</td>
<td></td>
</tr>
</tbody>
</table>
### Purpose

Configure SR-TE Policies

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RP/0/RSP0/CPU0:router(config-rpl)# if destination in (5.5.5.1/32) then</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-rpl-if)# set extcommunity color color1</strong></td>
</tr>
<tr>
<td><strong>RP/0/RSP0/CPU0:router(config-rpl-if)# endif</strong></td>
<td><strong>RP/0/RSP0/CPU0:router(config-rpl)# end-policy</strong></td>
</tr>
</tbody>
</table>

**Step 6**

**end-policy**

**Example:**

**RP/0/RSP0/CPU0:router(config-rpl)# end-policy**

**Ends the definition of a route policy and exits route policy configuration mode.**

**Step 7**

**router bgp as-number**

**Example:**

**RP/0/RSP0/CPU0:router(config)# router bgp 1**

**Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.**

**Step 8**

**bgp router-id ip-address**

**Example:**

**RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 10.10.0.2**

**Configures the local router with a specified router ID.**

**Step 9**

**address-family (ipv4 | ipv6) sr-policy**

**Example:**

**RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 sr-policy**

**Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.**

**Step 10**

**exit**

**Step 11**

**neighbor ip-address**

**Example:**

**RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.0.1**

**Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.**

**Step 12**

**remote-as as-number**

**Example:**

**RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 1**

**Creates a neighbor and assigns a remote autonomous system number to it.**
### Purpose

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 13</strong></td>
<td>Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>address-family ipv4 unicast</code></td>
</tr>
</tbody>
</table>

| **Step 14** | Applies the specified policy to IPv4 unicast routes. |
| **Example:** | `route-policy color out` |

| **Step 15** | Sends extended community attributes to external Border Gateway Protocol (eBGP) neighbors. |
| **Example:** | `send-extended-community-ebgp` |

---

### Setting CO Flag

The BGP-based steering mechanism matches BGP color and next-hop with that of an SR-TE policy. If the policy does not exist, BGP requests XTC to create an SR-TE policy with the associated color, end-point, and explicit paths. For color-only steering (NULL end-point), you can configure a color-only (CO) flag as part of the color extended community in BGP.

**Note**

See *Color-Only Steering, on page 50* for information about color-only steering (NULL end-point).

The behavior of the steering mechanism is based on the following values of the CO flags:

<table>
<thead>
<tr>
<th>co-flag 00</th>
<th>1. The BGP next-hop and color (&lt;N, C&gt;) is matched with an SR-TE policy of same (&lt;N, C&gt;).</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2. If a policy does not exist, then IGP path for the next-hop (N) is chosen.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>co-flag 01</th>
<th>1. The BGP next-hop and color (&lt;N, C&gt;) is matched with an SR-TE policy of same (&lt;N, C&gt;).</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2. If a policy does not exist, then an SR-TE policy with NULL end-point with the same address-family as (N) and color (C) is chosen.</td>
</tr>
<tr>
<td></td>
<td>3. If a policy with NULL end-point with same address-family as (N) does not exist, then an SR-TE policy with any NULL end-point and color (C) is chosen.</td>
</tr>
<tr>
<td></td>
<td>4. If no match is found, then IGP path for the next-hop (N) is chosen.</td>
</tr>
</tbody>
</table>
Configuration Example

Router(config)# extcommunity-set opaque overlay-color
Router(config-ext)# 1 co-flag 01
Router(config-ext)# end-set
Router(config)#
Router(config)# route-policy color
Router(config-rpl)# if destination in (5.5.5.1/32) then
Router(config-rpl-if)# set extcommunity color overlay-color
Router(config-rpl-if)# endif
Router(config-rpl)# pass
Router(config-rpl)# end-policy
Router(config)#

Using Binding Segments

The binding segment is a local segment identifying an SR-TE policy. Each SR-TE policy is associated with a binding segment ID (BSID). The BSID is a local label that is automatically allocated for each SR-TE policy when the SR-TE policy is instantiated.

Note

In Cisco IOS XR 6.3.2 and later releases, you can specify an explicit BSID for an SR-TE policy. See the following Explicit Binding SID section.

BSID can be used to steer traffic into the SR-TE policy and across domain borders, creating seamless end-to-end inter-domain SR-TE policies. Each domain controls its local SR-TE policies; local SR-TE policies can be validated and rerouted if needed, independent from the remote domain’s head-end. Using binding segments isolates the head-end from topology changes in the remote domain.

Packets received with a BSID as top label are steered into the SR-TE policy associated with the BSID. When the BSID label is popped, the SR-TE policy’s SID list is pushed.

BSID can be used in the following cases:

- Multi-Domain (inter-domain, inter-autonomous system)—BSIDs can be used to steer traffic across domain borders, creating seamless end-to-end inter-domain SR-TE policies.

- Large-Scale within a single domain—The head-end can use hierarchical SR-TE policies by nesting the end-to-end (edge-to-edge) SR-TE policy within another layer of SR-TE policies (aggregation-to-aggregation). The SR-TE policies are nested within another layer of policies using the BSIDs, resulting in seamless end-to-end SR-TE policies.

- Label stack compression—If the label-stack size required for an SR-TE policy exceeds the platform capability, the SR-TE policy can be seamlessly stitched to, or nested within, other SR-TE policies using a binding segment.

- BGP SR-TE Dynamic—The head-end steers the packet into a BGP-based FIB entry whose next hop is a binding-SID.
Explicit Binding SID

Use the `binding-sid explicit {fallback-dynamic | enforce-srlb}` command to request that the SR-TE policy uses a BSID value that you provide. Explicit BSIDs are allocated from the segment routing local block (SRLB) or the dynamic range of labels.

A best-effort is made to request and obtain this BSID for the SR-TE policy. If requested BSID is not available (if it does not fall within the available SRLB or is already used by another application or SR-TE policy), the policy stays down.

You can specify how the BSID allocation behaves if the BSID value is not available:

- Fallback to dynamic allocation – If the BSID is not available, the BSID is allocated dynamically and the policy comes up:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# binding-sid explicit fallback-dynamic
```

- Strict SRLB enforcement – If the BSID is not within the SRLB, the policy stays down:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# binding-sid explicit enforce-srlb
```

Stitching SR-TE Polices Using Binding SID: Example

In this intra-domain example, three SR-TE policies are stitched together to form a seamless end-to-end path from node 1 to node 10.

**Figure 3: Intra-Domain Topology**

**Step 1** Configure an SR-TE policy on node 5 to node 10 via node 9. Node 5 automatically allocates a binding-SID (24012) for the SR-TE policy.

**Example:**

```
RP/0/0/CPU0:xrvr-5(config)# explicit-path name PATH5-9_10
```
Configure SR-TE Policies

Step 2 Configure an SR-TE policy on node 3 to node 5 via node 4 and Link4-6, and push the binding-SID of the SR-TE policy at node 5 (24012) to stitch to the SR-TE policy on node 5. Node 3 automatically allocates a binding-SID (24008) for this SR-TE policy.

Example:

```
RP/0/0/CPU0:xrvr-3(config)# explicit-path name PATH4_4-6_5_BSID
RP/0/0/CPU0:xrvr-3(config-expl-path)# index 10 next-address strict ipv4 unicast 10.1.1.4
RP/0/0/CPU0:xrvr-3(config-expl-path)# index 20 next-address strict ipv4 unicast 192.168.46.6
RP/0/0/CPU0:xrvr-3(config-expl-path)# index 30 next-address strict ipv4 unicast 10.1.1.5
RP/0/0/CPU0:xrvr-3(config-expl-path)# index 40 next-label 24012
RP/0/0/CPU0:xrvr-3(config-expl-path)# exit
RP/0/0/CPU0:xrvr-3(config-if)# interface tunnel-te1
RP/0/0/CPU0:xrvr-3(config-if)# ipv4 unnumbered Loopback0
RP/0/0/CPU0:xrvr-3(config-if)# destination 10.1.1.10
RP/0/0/CPU0:xrvr-3(config-if)# path-option 1 explicit name PATH4_4-6_5_BSID segment-routing
RP/0/0/CPU0:xrvr-3(config-if)# commit
RP/0/0/CPU0:xrvr-3# show mpls traffic-eng tunnels 1 detail
Name: tunnel-te1 Destination: 10.1.1.10 Ifhandle:0x780
  Signalled-Name: xrvr-3_t1
  Status:
    Admin: up Oper: up Path: valid Signalling: connected
    path option 1, (Segment-Routing) type explicit (Path Weight 10)
  Binding SID: 24008
  ...
  Segment-Routing Path Info (IS-IS 1 level-2)
    Segment0[Link]: 10.1.1.4-192.168.46.6, Label: 24003
    Segment1[Node]: 10.1.1.5, Label: 16005
    Segment2[Node]: 10.1.1.5, Label: 16005
    Segment3[ - ]: Label: 24012
```

Step 3 Configure an SR-TE policy on node 1 to node 3 and push the binding-SID of the SR-TE policy at node 3 (24008) to stitch to the SR-TE policy on node 3.
**Example:**

```
RP/0/0/CPU0:xrvr-1(config)# explicit-path name PATH3_BSID
RP/0/0/CPU0:xrvr-1(config-expl-path)# index 10 next-address strict ipv4 unicast 10.1.1.3
RP/0/0/CPU0:xrvr-1(config-expl-path)# index 20 next-label 24008
RP/0/0/CPU0:xrvr-1(config-expl-path)# exit

RP/0/0/CPU0:xrvr-1(config)# interface tunnel-te1
RP/0/0/CPU0:xrvr-1(config-if)# ipv4 unnumbered Loopback0
RP/0/0/CPU0:xrvr-1(config-if)# destination 10.1.1.10
RP/0/0/CPU0:xrvr-1(config-if)# path-option 1 explicit name PATH3_BSID segment-routing
RP/0/0/CPU0:xrvr-1(config-if)# commit

RP/0/0/CPU0:xrvr-1# show mpls traffic-eng tunnels 1 detail
Name: tunnel-te1 Destination: 10.1.1.10 Ifhandle:0x2f80
  Signalled-Name: xrvr-1_t1
  Status:
    Admin: up Oper: up Path: valid Signalling: connected
    path option 1, (Segment-Routing) type explicit PATH3_BSID (Basis for Setup)
  Binding SID: 24002
  Segment-Routing Path Info (IS-IS 1 level-2)
    Segment0[Node]: 10.1.1.3, Label: 16003
    Segment1[-]: Label: 24008
```

The path is a chain of SR-TE policies stitched together using the binding-SIDs, providing a seamless end-to-end path.

```
RP/0/0/CPU0:xrvr-1# traceroute 10.1.1.10
Type escape sequence to abort.
Tracing the route to 10.1.1.10
1 99.1.2.2 [MPLS: Labels 16003/24008 Exp 0] 29 msec 19 msec 19 msec
2 99.2.3.3 [MPLS: Label 24008 Exp 0] 29 msec 19 msec 19 msec
3 99.3.4.4 [MPLS: Labels 24003/16005/24012 Exp 0] 29 msec 19 msec 19 msec
4 99.4.6.6 [MPLS: Labels 16005/24012 Exp 0] 29 msec 29 msec 19 msec
5 99.5.6.6 [MPLS: Label 24012 Exp 0] 29 msec 29 msec 19 msec
6 99.5.9.9 [MPLS: Label 16010 Exp 0] 19 msec 19 msec 19 msec
7 99.9.10.10 29 msec 19 msec 19 msec
```
CHAPTER 8

Configure Segment Routing Path Computation Element

The Segment Routing Path Computation Element (SR-PCE) provides stateful PCE functionality by extending the existing IOS-XR PCEP functionality with additional capabilities. SR-PCE is supported on the MPLS data plane and IPv4 control plane.

Note
To install SR-PCE, you need to install an instance of Cisco IOS XRv 9000 Router. Refer to the Cisco IOS XRv 9000 Router Installation and Configuration Guide for more information.

• About SR-PCE, on page 65
• Configure SR-PCE, on page 66
• PCE-Initiated SR Policies for Traffic Management, on page 69

About SR-PCE

The path computation element protocol (PCEP) describes a set of procedures by which a path computation client (PCC) can report and delegate control of head-end label switched paths (LSPs) sourced from the PCC to a PCE peer. The PCE can request the PCC to update and modify parameters of LSPs it controls. The stateful model also enables a PCC to allow the PCE to initiate computations allowing the PCE to perform network-wide orchestration.

Note
For more information on PCE, PCC, and PCEP, refer to the Path Computation Element section in the MPLS Configuration Guide for Cisco ASR 9000 Series Routers.

SR-PCE learns topology information by way of IGP (OSPF or IS-IS) or through BGP Link-State (BGP-LS). SR-PCE is capable of computing paths using the following methods:

• TE metric—SR-PCE uses the TE metric in its path calculations to optimize cumulative TE metric.
• IGP metric—SR-PCE uses the IGP metric in its path calculations to optimize reachability.
• LSP Disjointness—SR-PCE uses the path computation algorithms to compute a pair of disjoint LSPs. The disjoint paths can originate from the same head-end or different head-ends. Disjoint level refers to
the type of resources that should not be shared by the two computed paths. SR-PCE supports the following disjoint path computations:

- Link – Specifies that links are not shared on the computed paths.
- Node – Specifies that nodes are not shared on the computed paths.
- SRLG – Specifies that links with the same SRLG value are not shared on the computed paths.
- SRLG-node – Specifies that SRLG and nodes are not shared on the computed paths.

When the first request is received with a given disjoint-group ID, the first LSP is computed, encoding the shortest path from the first source to the first destination. When the second LSP request is received with the same disjoint-group ID, information received in both requests is used to compute two disjoint paths: one path from the first source to the first destination, and another path from the second source to the second destination. Both paths are computed at the same time.

## Configure SR-PCE

This task explains how to configure SR-PCE.

**Before you begin**

Optionally install and configure an instance of Cisco IOS XRv 9000 Router.

### SUMMARY STEPS

1. `configure`
2. `pce`
3. `address ipv4 address`
4. `state-sync ipv4 address`
5. `tcp-buffer size`
6. `password {clear | encrypted} password`
7. `segment-routing {strict-sid-only | te-latency}`
8. `timers`
9. `keepalive time`
10. `minimum-peer-keepalive time`
11. `reoptimization time`
12. `exit`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>pce</td>
</tr>
</tbody>
</table>

Example:
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>RP/0/RSP0/CPU0:router(config)# pce</code></td>
<td><strong>Purpose</strong></td>
</tr>
</tbody>
</table>

**Step 3**

`address ipv4 address`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce)# address ipv4 192.168.0.1
```

Configures a PCE IPv4 address.

**Step 4**

`state-sync ipv4 address`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce)# state-sync ipv4 192.168.0.3
```

Configures the remote peer for state synchronization.

**Step 5**

`tcp-buffer size`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce)# tcp-buffer 1024000
```

Configures the transmit and receive TCP buffer size for each PCEP session, in bytes. The default buffer size is 256000. The valid range is from 204800 to 1024000.

**Step 6**

`password {clear | encrypted} password`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce)# password encrypted pwd1
```

Enables TCP authentication for all PCEP peers. Any TCP segment coming from the PCC that does not contain a MAC matching the configured password will be rejected. Specify if the password is encrypted or clear text.

**Step 7**

`segment-routing {strict-sid-only | te-latency}`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce)# segment-routing strict-sid-only
```

Configures the segment routing algorithm to use strict SID or TE latency.

**Note**

This setting is global and applies to all LSPs that request a path from this controller.

**Step 8**

`timers`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce)# timers
```

Enters timer configuration mode.

**Step 9**

`keepalive time`

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce-timers)# keepalive 60
```

Configures the timer value for locally generated keep-alive messages. The default time is 30 seconds.
Configure Segment Routing Path Computation Element

Configure the Disjoint Policy (Optional)

This task explains how to configure the SR-PCE to compute disjointness for a pair of LSPs signaled by PCCs that do not include the PCEP association group-ID object in their PCEP request. This can be beneficial for deployments where PCCs do not support this PCEP object or when the network operator prefers to manage the LSP disjoint configuration centrally.

### SUMMARY STEPS

1. disjoint-path
2. group-id value type {link | node | srlg | srlg-node} [sub-id value]
3. strict
4. lsp {1 | 2} pcc ipv4 address lsp-name lsp_name [shortest-path]

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 disjoint-path</td>
<td>Enters disjoint configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# disjoint-path</td>
<td></td>
</tr>
<tr>
<td>Step 2 group-id value type {link</td>
<td>node</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
</tbody>
</table>
\[ \text{Purpose} \] Command or Action | Purpose
---|---
• link—Specifies that links are not shared on the computed paths.
• node—Specifies that nodes are not shared on the computed paths.
• srlg—Specifies that links with the same SRLG value are not shared on the computed paths.
• srlg-node—Specifies that SRLG and nodes are not shared on the computed paths.

If a pair of paths that meet the requested disjointness level cannot be found, then the paths will automatically fallback to a lower level:
• If the requested disjointness level is SRLG or node, then link-disjoint paths will be computed.
• If the requested disjointness level was link, or if the first fallback from SRLG or node disjointness failed, then the lists of segments encoding two shortest paths, without any disjointness constraint, will be computed.

**Step 3** \( \text{strict} \)

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce-disjoint)# strict
```

(Optional) Prevents the automatic fallback behavior of the preferred level of disjointness. If a pair of paths that meet the requested disjointness level cannot be found, the disjoint calculation terminates and no new path is provided. The existing path is not modified.

**Step 4** \( \text{lsp \{1 | 2\} pcc ipv4 address lsp-name lsp_name [shortest-path]} \)

**Example:**

```
RP/0/RSP0/CPU0:router(config-pce-disjoint)# lsp 1 pcc ipv4 192.168.0.1 lsp-name rtrA_t1 shortest-path
```

Adds LSPs to the disjoint group.

The **shortest-path** keyword forces one of the disjoint paths to follow the shortest path from the source to the destination. This option can only be applied to the first LSP specified.

**PCE-Initiated SR Policies for Traffic Management**

An SR-TE policy can be configured on the path computation element (PCE) to reduce link congestion or to minimize the number of network touch points.
PCE-Initiated SR Policies for Traffic Management

The PCE-initiated SR-TE policies are entered in PCE configuration mode. For more information on configuring SR-TE policies, see the Configure SR-TE Policies, on page 47.

The PCE collects network information, such as traffic demand and link utilization. When the PCE determines that a link is congested, it identifies one or more flows that are causing the congestion. The PCE finds a suitable path and deploys an SR-TE policy to divert those flows, without moving the congestion to another part of the network. When there is no more link congestion, the policy is removed.

To minimize the number of network touch points, an application, such as a Network Services Orchestrator (NSO), can request the PCE to create an SR-TE policy. PCE deploys the SR-TE policy using PCC-PCE communication protocol (PCEP).

1. PCE sends a PCInitiate message to the PCC.
2. If the PCInitiate message is valid, the PCC sends a PCRpt message; otherwise, it sends PCErr message.
3. If the PCInitiate message is accepted, the PCE updates the SR-TE policy by sending PCUpd message.

You can achieve high-availability by configuring multiple PCEs with SR-TE policies. If the head-end (PCC) loses connectivity with one PCE, another PCE can assume control of the SR-TE policy.

Configuration Example

To configure a PCE-initiated SR-TE policy, you must complete the following configurations:

1. Enter PCE configuration mode.
2. Create the segment list.
3. Create the policy.

/* Enter PCE configuration mode and create the SR-TE segment lists */
Router# configure
Router(config)# pce

/* Create the SR-TE segment lists */
Router(config-pce)# segment-routing
Router(config-pce-sr)# traffic-eng
Router(config-pce-sr-te)# segment-list name addr2a
Router(config-pce-sr-te-sl)# index 1 address ipv4 14.14.14.4
Router(config-pce-sr-te-sl)# exit

/* Create the SR-TE policy */
Router(config-pce-sr-te)# peer ipv4 1.1.1.1
Router(config-pce-sr-te)# policy P1
Router(config-pce-sr-te-policy)# color 2 end-point ipv4 2.2.2.2
Router(config-pce-sr-te-policy)# candidate-paths
Router(config-pce-sr-te-policy-path)# preference 50
Router(config-pce-sr-te-pp-index)# explicit segment-list addr2a
Router(config-pce-sr-te-pp-info)# end
Router(config)#
Running Config

```
pce
  segment-routing
  traffic-eng
  segment-list name addr2a
    index 1 address ipv4 14.14.14.4
  !
  peer ipv4 1.1.1.1
  policy P1
  color 2 end-point ipv4 2.2.2.2
  candidate-paths
  preference 50
  explicit segment-list addr2a
  !
  
```
PCE-Initiated SR Policies for Traffic Management
Configure Topology-Independent Loop-Free Alternate (TI-LFA)

Topology-Independent Loop-Free Alternate (TI-LFA) uses segment routing to provide link, node, and Shared Risk Link Groups (SRLG) protection in topologies where other fast reroute techniques cannot provide protection. The goal of TI-LFA is to reduce the packet loss that results while routers converge after a topology change due to a link failure. Rapid failure repair (< 50 msec) is achieved through the use of pre-calculated backup paths that are loop-free and safe to use until the distributed network convergence process is completed.

Note

TI-LFA supports IPv4 only.

TI-LFA provides link protection. The link is excluded during the post convergence backup path calculation.

TI-LFA node protection provides protection from node failures. The neighbor node is excluded during the post convergence backup path calculation.

Shared Risk Link Groups (SRLG) refer to situations in which links in a network share a common fiber (or a common physical attribute). These links have a shared risk: when one link fails, other links in the group might also fail. TI-LFA SRLG protection attempts to find the post-convergence backup path that excludes the SRLG of the protected link. All local links that share any SRLG with the protecting link are excluded.

When you enable link protection, you can also enable node protection, SRLG protection, or both, and specify a tiebreaker priority in case there are multiple LFAs.

For IS-IS, TI-LFA node protection and SRLG protection can be configured on the interface or the instance. For OSPF, TI-LFA node protection and SRLG protection are configured on the interface.

- Configuring TI-LFA for IS-IS, on page 73
- Configuring TI-LFA for OSPF, on page 75
- TI-LFA Node and SRLG Protection: Examples, on page 77
- Configuring Global Weighted SRLG Protection, on page 78

Configuring TI-LFA for IS-IS

This task describes how to enable per-prefix Topology Independent Loop-Free Alternate (TI-LFA) computation to converge traffic flows around link, node, and SRLG failures.
**Before you begin**

Ensure that the following topology requirements are met:

- Router interfaces are configured as per the topology.
- Routers are configured with IS-IS.
- Segment routing LSPs are configured.

**SUMMARY STEPS**

1. `configure`
2. `router isis instance-id`
3. `interface type interface-path-id`
4. `address-family ipv4 [unicast]`
5. `fast-reroute per-prefix`
6. `fast-reroute per-prefix ti-lfa`
7. `fast-reroute per-prefix tiebreaker {node-protecting | srlg-disjoint} index priority`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><code>configure</code></td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
</tbody>
</table>
| Step 2 | `router isis instance-id`  
**Example:**  
`RP/0/RSP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.  
**Note** You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command. |
| Step 3 | `interface type interface-path-id`  
**Example:**  
`RP/0/RSP0/CPU0:router(config-isis)# interface GigabitEthernet0/0/2/1`

`RP/0/RSP0/CPU0:router(config-isis)# interface Bundle-Ether1` | Enters interface configuration mode. |
| Step 4 | `address-family ipv4 [unicast]`  
**Example:**  
`RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast` | Specifies the IPv4 address family, and enters router address family configuration mode. |
| Step 5 | `fast-reroute per-prefix`  
**Example:** | Enables per-prefix fast reroute. |
### Purpose

**Purpose**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>RP/0/RSP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix</code></td>
<td>Enables per-prefix TI-LFA fast reroute link protection.</td>
</tr>
</tbody>
</table>

**Step 6**

**fast-reroute per-prefix ti-lfa**

**Example:**

```
RP/0/RSP0/CPU0:router(config-isis-if-af)#
fast-reroute per-prefix ti-lfa
```

**Step 7**

**fast-reroute per-prefix tiebreaker { node-protecting | srlg-disjoint } index priority**

**Example:**

```
RP/0/RSP0/CPU0:router(config-isis-if-af)#
fast-reroute per-prefix srlg-disjoint index 100
```

- The lower the `priority` value, the higher the priority of the rule. Link protection always has a lower priority than node or SRLG protection.

**Note**

TI-LFA has been successfully configured for segment routing.

## Configuring TI-LFA for OSPF

This task describes how to enable per-prefix Topology Independent Loop-Free Alternate (TI-LFA) computation to converge traffic flows around link, node, and SRLG failures.

### Note

TI-LFA can be configured on the instance, area, or interface. When configured on the instance or area, all interfaces in the instance or area inherit the configuration.

### Before you begin

Ensure that the following topology requirements are met:

- Router interfaces are configured as per the topology.
- Routers are configured with OSPF.
- Segment routing LSPs are configured.

### SUMMARY STEPS

1. `configure`
2. `router ospf process-name`
3. `area area-id`
4. `interface type interface-path-id`
5. `fast-reroute per-prefix`
**Configure Topology-Independent Loop-Free Alternate (TI-LFA)**

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enables OSPF routing for the specified routing process, and places the router in router configuration mode.</td>
</tr>
</tbody>
</table>
| **Step 2** `router ospf process-name`  
  Example:  
  ```bash
  RP/0/RSP0/CPU0:router(config)# router ospf 1
  ``` | Enters area configuration mode. |
| **Step 3** `area area-id`  
  Example:  
  ```bash
  RP/0/RSP0/CPU0:router(config-ospf)# area 1
  ``` | Enters interface configuration mode. |
| **Step 4** `interface type interface-path-id`  
  Example:  
  ```bash
  RP/0/RSP0/CPU0:router(config-ospf-ar)# interface GigabitEthernet0/0/2/1
  ``` | Enables per-prefix fast reroute. |
| **Step 5** `fast-reroute per-prefix`  
  Example:  
  ```bash
  RP/0/RSP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix
  ``` | Enables per-prefix TI-LFA fast reroute link protection. |
| **Step 6** `fast-reroute per-prefix ti-lfa`  
  Example:  
  ```bash
  RP/0/RSP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix ti-lfa
  ``` | Enables TI-LFA node or SRLG protection and specifies the tiebreaker priority. Valid priority values are from 1 to 255. The lower the priority value, the higher the priority of the rule. Link protection always has a lower priority than node or SRLG protection. |
| **Step 7** `fast-reroute per-prefix tiebreaker { node-protecting | srlg-disjoint } index priority`  
  Example:  
  ```bash
  RP/0/RSP0/CPU0:router(config-isis-ar-if)# fast-reroute per-prefix srlg-disjoint index 100
  ``` | The same attribute cannot be configured more than once on an interface. |

TI-LFA has been successfully configured for segment routing.
TI-LFA Node and SRLG Protection: Examples

The following examples show the configuration of the tiebreaker priority for TI-LFA node and SRLG protection, and the behavior of post-convergence backup-path. These examples use OSPF, but the same configuration and behavior applies to IS-IS.

Example: Enable link-protecting and node-protecting TI-LFA

```
router ospf 1
area 1
    interface GigabitEthernet0/0/2/1
        fast-reroute per-prefix
        fast-reroute per-prefix ti-lfa
        fast-reroute per-prefix tiebreaker node-protecting index 100
```

Both link-protecting and node-protecting TI-LFA backup paths will be computed. If the priority associated with the node-protecting tiebreaker is higher than any other tiebreakers, then node-protecting post-convergence backup paths will be selected, if it is available.

Example: Enable link-protecting and SRLG-protecting TI-LFA

```
router ospf 1
area 1
    interface GigabitEthernet0/0/2/1
        fast-reroute per-prefix
        fast-reroute per-prefix ti-lfa
        fast-reroute per-prefix tiebreaker srlg-disjoint index 100
```

Both link-protecting and SRLG-protecting TI-LFA backup paths will be computed. If the priority associated with the SRLG-protecting tiebreaker is higher than any other tiebreakers, then SRLG-protecting post-convergence backup paths will be selected, if it is available.

Example: Enable link-protecting, node-protecting and SRLG-protecting TI-LFA

```
router ospf 1
area 1
    interface GigabitEthernet0/0/2/1
        fast-reroute per-prefix
        fast-reroute per-prefix ti-lfa
        fast-reroute per-prefix tiebreaker node-protecting index 100
        fast-reroute per-prefix tiebreaker srlg-disjoint index 200
```

Link-protecting, node-protecting, and SRLG-protecting TI-LFA backup paths will be computed. If the priority associated with the node-protecting tiebreaker is highest from all tiebreakers, then node-protecting post-convergence backup paths will be selected, if it is available. If the node-protecting backup path is not available, SRLG-protecting post-convergence backup path will be used, if it is available.
Configuring Global Weighted SRLG Protection

A shared risk link group (SRLG) is a set of links sharing a common resource and thus shares the same risk of failure. The existing loop-free alternate (LFA) implementations in interior gateway protocols (IGPs) support SRLG protection. However, the existing implementation considers only the directly connected links while computing the backup path. Hence, SRLG protection may fail if a link that is not directly connected but shares the same SRLG is included while computing the backup path. Global weighted SRLG protection feature provides better path selection for the SRLG by associating a weight with the SRLG value and using the weights of the SRLG values while computing the backup path.

To support global weighted SRLG protection, you need information about SRLGs on all links in the area topology. You can flood SRLGs for remote links using ISIS or manually configuring SRLGs on remote links.

Configuration Examples: Global Weighted SRLG Protection

There are three types of configurations that are supported for the global weighted SRLG protection feature.

- local SRLG with global weighted SRLG protection
- remote SRLG flooding
- remote SRLG static provisioning

This example shows how to configure the local SRLG with global weighted SRLG protection feature.

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg)# name group value 100
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection weighted-global
```

This example shows how to configure the global weighted SRLG protection feature with remote SRLG flooding. The configuration includes local and remote router configuration. On the local router, the global weighted SRLG protection is enabled by using the `fast-reroute per-prefix srlg-protection weighted-global` command. In the remote router configuration, you can control the SRLG value flooding by using the `advertise application lfa link-attributes srlg` command. You should also globally configure SRLG on the remote router.

The local router configuration for global weighted SRLG protection with remote SRLG flooding is as follows:

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/1
RP/0/RP0/CPU0:router(config-srlg)# name group1
RP/0/RP0/CPU0:router(config-srlg)# admin-weight 5000
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection weighted-global
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint index 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix point-to-point
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg
```

This example shows how to configure the global weighted SRLG protection feature with remote SRLG flooding. The configuration includes local and remote router configuration. On the local router, the global weighted SRLG protection is enabled by using the `fast-reroute per-prefix srlg-protection weighted-global` command. In the remote router configuration, you can control the SRLG value flooding by using the `advertise application lfa link-attributes srlg` command. You should also globally configure SRLG on the remote router.

The local router configuration for global weighted SRLG protection with remote SRLG flooding is as follows:
Configure Topology-Independent Loop-Free Alternate (TI-LFA)  

Configuring Global Weighted SRLG Protection

The remote router configuration for global weighted SRLG protection with remote SRLG flooding is as follows:

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# name group value 100
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection weighted-global
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint index 1
RP/0/RP0/CPU0:router(config-isis-if-af)# exit
RP/0/RP0/CPU0:router(config-isis)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis-if-af)# exit
RP/0/RP0/CPU0:router(config-isis)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
```

This example shows configuring the global weighted SRLG protection feature with static provisioning of SRLG values for remote links. You should perform these configurations on the local router.

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# name group value 100
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection weighted-global
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint index 1
RP/0/RP0/CPU0:router(config-isis-if-af)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis-if-af)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.1 next-hop ipv4 address 10.0.4.2
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.2 next-hop ipv4 address 10.0.4.1
```
Configure Topology-Independent Loop-Free Alternate (TI-LFA)
CHAPTER 10

Configure Segment Routing Microloop Avoidance

The Segment Routing Microloop Avoidance feature enables link-state routing protocols, such as IS-IS and OSPF, to prevent or avoid microloops during network convergence after a topology change.

- About Segment Routing Microloop Avoidance, on page 81
- Configure Segment Routing Microloop Avoidance for IS-IS, on page 81
- Configure Segment Routing Microloop Avoidance for OSPF, on page 82

About Segment Routing Microloop Avoidance

Microloops are brief packet loops that occur in the network following a topology change (link down, link up, or metric change events). Microloops are caused by the non-simultaneous convergence of different nodes in the network. If nodes converge and send traffic to a neighbor node that has not converged yet, traffic may be looped between these two nodes, resulting in packet loss, jitter, and out-of-order packets.

The Segment Routing Microloop Avoidance feature detects if microloops are possible following a topology change. If a node computes that a microloop could occur on the new topology, the node creates a loop-free SR-TE policy path to the destination using a list of segments. After the RIB update delay timer expires, the SR-TE policy is replaced with regular forwarding paths.

Configure Segment Routing Microloop Avoidance for IS-IS

This task describes how to enable Segment Routing Microloop Avoidance and set the Routing Information Base (RIB) update delay value for IS-IS.

Before you begin

Ensure that the following topology requirements are met:

- Router interfaces are configured as per the topology.
- Routers are configured with IS-IS.
- Segment routing for IS-IS is configured.

SUMMARY STEPS

1. configure
2. `router isis instance-id`
3. `address-family ipv4 [ unicast ]`
4. `microloop avoidance segment-routing`
5. `microloop avoidance rib-update-delay delay-time`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><code>configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><code>router isis instance-id</code></td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode. You can change the level of routing to be performed by a particular routing instance by using the <code>is-type router</code> configuration command.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config)# router isis 1</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><code>address-family ipv4 [ unicast ]</code></td>
<td>Specifies the IPv4 address family and enters router address family configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config-isis)# address-family ipv4 unicast</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><code>microloop avoidance segment-routing</code></td>
<td>Enables Segment Routing Microloop Avoidance.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config-isis-af)# microloop avoidance segment-routing</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td><code>microloop avoidance rib-update-delay delay-time</code></td>
<td>Specifies the amount of time the node uses the microloop avoidance policy before updating its forwarding table. The <code>delay-time</code> is in milliseconds. The range is from 1-60000. The default value is 5000.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config-isis-af)# microloop avoidance rib-update-delay 3000</code></td>
<td></td>
</tr>
</tbody>
</table>

---

**Configure Segment Routing Microloop Avoidance for OSPF**

This task describes how to enable Segment Routing Microloop Avoidance and set the Routing Information Base (RIB) update delay value for OSPF.

**Before you begin**

Ensure that the following topology requirements are met:

- Router interfaces are configured as per the topology.
- Routers are configured with OSPF.
- Segment routing for OSPF is configured.
### SUMMARY STEPS

1. `configure`
2. `router ospf process-name`
3. `microloop avoidance segment-routing`
4. `microloop avoidance rib-update-delay delay-time`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><code>configure</code></td>
<td></td>
</tr>
</tbody>
</table>
| Step 2 | `router ospf process-name`  
Example:  
`RP/0/RSP0/CPU0:router(config)# router ospf 1` | Enables OSPF routing for the specified routing process, and places the router in router configuration mode. |
| Step 3 | `microloop avoidance segment-routing`  
Example:  
`RP/0/RSP0/CPU0:router(config-ospf)# microloop avoidance segment-routing` | Enables Segment Routing Microloop Avoidance. |
| Step 4 | `microloop avoidance rib-update-delay delay-time`  
Example:  
`RP/0/RSP0/CPU0:router(config-ospf)# microloop avoidance rib-update-delay 3000` | Specifies the amount of time the node uses the microloop avoidance policy before updating its forwarding table. The `delay-time` is in milliseconds. The range is from 1-60000. The default value is 5000. |
CHAPTER 11

Configure Segment Routing Mapping Server

The mapping server is a key component of the interworking between LDP and segment routing. It enables SR-capable nodes to interwork with LDP nodes. The mapping server advertises Prefix-to-SID mappings in IGP on behalf of other non-SR-capable nodes.

- Segment Routing Mapping Server, on page 85
- Segment Routing and LDP Interoperability, on page 86
- Configuring Mapping Server, on page 88
- Enable Mapping Advertisement, on page 90
- Enable Mapping Client, on page 92

Segment Routing Mapping Server

The mapping server functionality in Cisco IOS XR segment routing centrally assigns prefix-SIDs for some or all of the known prefixes. A router must be able to act as a mapping server, a mapping client, or both.

- A router that acts as a mapping server allows the user to configure SID mapping entries to specify the prefix-SIDs for some or all prefixes. This creates the local SID-mapping policy. The local SID-mapping policy contains non-overlapping SID-mapping entries. The mapping server advertises the local SID-mapping policy to the mapping clients.

- A router that acts as a mapping client receives and parses remotely received SIDs from the mapping server to create remote SID-mapping entries.

- A router that acts as a mapping server and mapping client uses the remotely learnt and locally configured mapping entries to construct the non-overlapping consistent active mapping policy. IGP instance uses the active mapping policy to calculate the prefix-SIDs of some or all prefixes.

The mapping server automatically manages the insertions and deletions of mapping entries to always yield an active mapping policy that contains non-overlapping consistent SID-mapping entries.

- Locally configured mapping entries must not overlap each other.

- The mapping server takes the locally configured mapping policy, as well as remotely learned mapping entries from a particular IGP instance, as input, and selects a single mapping entry among overlapping mapping entries according to the preference rules for that IGP instance. The result is an active mapping policy that consists of non-overlapping consistent mapping entries.

- At steady state, all routers, at least in the same area or level, must have identical active mapping policies.
Segment Routing Mapping Server Restrictions

- The position of the mapping server in the network is not important. However, since the mapping advertisements are distributed in IGP using the regular IGP advertisement mechanism, the mapping server needs an IGP adjacency to the network.

- The role of the mapping server is crucial. For redundancy purposes, you should configure multiple mapping servers in the networks.

- The mapping server functionality does not support a scenario where SID-mapping entries learned through one IS-IS instance are used by another IS-IS instance to determine the prefix-SID of a prefix. For example, mapping entries learnt from remote routers by ‘router isis 1’ cannot be used to calculate prefix-SIDs for prefixes learnt, advertised, or downloaded to FIB by ‘router isis 2’. A mapping server is required for each IS-IS area.

- Segment Routing Mapping Server does not support Virtual Routing and Forwarding (VRF) currently.

Segment Routing and LDP Interoperability

IGP provides mechanisms through which segment routing (SR) interoperate with label distribution protocol (LDP). The control plane of segment routing co-exists with LDP.

The Segment Routing Mapping Server (SRMS) functionality in SR is used to advertise SIDs for destinations, in the LDP part of the network, that do not support SR. SRMS maintains and advertises segment identifier (SID) mapping entries for such destinations. IGP propagates the SRMS mapping entries and interacts with SRMS to determine the SID value when programming the forwarding plane. IGP installs prefixes and corresponding labels, into routing information base (RIB), that are used to program the forwarding information base (FIB).

Example: Segment Routing LDP Interoperability

Consider a network with a mix of segment routing (SR) and label distribution protocol (LDP). A continuous multiprotocol label switching (MPLS) LSP (Labeled Switched Path) can be established by facilitating interoperability. One or more nodes in the SR domain act as segment routing mapping server (SRMS). SRMS advertises SID mappings on behalf of non-SR capable nodes. Each SR-capable node learns about SID assigned to non-SR capable nodes without explicitly configuring individual nodes.

Consider a network as shown in the following image. This network is a mix of both LDP and SR-capable nodes.

In this mixed network:

- Nodes P6, P7, P8, PE4 and PE3 are LDP-capable
- Nodes PE1, PE2, P5 and P6 are SR-capable
- Nodes PE1, PE2, P5 and P6 are configured with segment routing global block (SRGB) of (100, 200)
- Nodes PE1, PE2, P5 and P6 are configured with node segments of 101, 102, 105 and 106 respectively
A service flow must be established from PE1 to PE3 over a continuous MPLS tunnel. This requires SR and LDP to interoperate.

**LDP to SR**

The traffic flow from LDP to SR (right to left) involves:

1. PE3 learns a service route whose nhop is PE1. PE3 has an LDP label binding from the nhop P8 for the FEC PE1. PE3 forwards the packet P8.
2. P8 has an LDP label binding from its nhop P7 for the FEC PE1. P8 forwards the packet to P7.
3. P7 has an LDP label binding from its nhop P6 for the FEC PE1. P7 forwards the packet to P6.
4. P6 does not have an LDP binding from its nhop P5 for the FEC PE1. But P6 has an SR node segment to the IGP route PE1. P6 forwards the packet to P5 and swaps its local LDP label for FEC PE1 by the equivalent node segment 101. This process is called label merging.
5. P5 pops 101, assuming PE1 has advertised its node segment 101 with the penultimate-pop flag set and forwards to PE1.
6. PE1 receives the tunneled packet and processes the service label.

The end-to-end MPLS tunnel is established from an LDP LSP from PE3 to P6 and the related node segment from P6 to PE1.

**SR to LDP**

Suppose that the operator configures P5 as a Segment Routing Mapping Server (SRMS) and advertises the mappings (P7, 107), (P8, 108), (PE3, 103) and (PE4, 104). If PE3 was SR-capable, the operator may have configured PE3 with node segment 103. Because PE3 is non-SR capable, the operator configures that policy at the SRMS; the SRMS advertises the mapping on behalf of the non-SR capable nodes. Multiple SRMS servers can be provisioned in a network for redundancy. The mapping server advertisements are only understood by the SR-capable nodes. The SR capable routers install the related node segments in the MPLS data plane in exactly the same manner if node segments were advertised by the nodes themselves.

The traffic flow from SR to LDP (left to right) involves:

1. PE1 installs the node segment 103 with nhop P5 in exactly the same manner if PE3 had advertised node segment 103.
2. P5 swaps 103 for 103 and forwards to P6.
3. The nhop for P6 for the IGP route PE3 is non-SR capable. (P7 does not advertise the SR capability.) However, P6 has an LDP label binding from that nhop for the same FEC. (For example, LDP label 1037.) P6 swaps 103 for 1037 and forwards to P7. We refer to this process as label merging.
4. P7 swaps this label with the LDP label received from P8 and forwards to P8.
5. P8 pops the LDP label and forwards to PE3.
6. PE3 receives the packet and processes as required.

The end-to-end MPLS LSP is established from an SR node segment from PE1 to P6 and an LDP LSP from P6 to PE3.
Configuring Mapping Server

Perform these tasks to configure the mapping server and to add prefix-SID mapping entries in the active local mapping policy.

SUMMARY STEPS

1. configure
2. segment-routing
3. mapping-server
4. prefix-sid-map
5. address-family ipv4 | ipv6
6. ip-address/prefix-length first-SID-value range
7. commit
8. show segment-routing mapping-server prefix-sid-map [ipv4 | ipv6] [detail]

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>configure</td>
</tr>
<tr>
<td>Step 2</td>
<td>segment-routing</td>
</tr>
<tr>
<td>Example:</td>
<td>Enables segment routing.</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# segment-routing</td>
</tr>
<tr>
<td>Step 3</td>
<td>mapping-server</td>
</tr>
<tr>
<td>Example:</td>
<td>Enables mapping server configuration mode.</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-sr)# mapping-server</td>
</tr>
<tr>
<td>Step 4</td>
<td>prefix-sid-map</td>
</tr>
<tr>
<td>Example:</td>
<td>Enables prefix-SID mapping configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Note Two-way prefix SID can be enabled directly under IS-IS or through a mapping server.</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-sr-ms)# prefix-sid-map</td>
</tr>
<tr>
<td>Step 5</td>
<td>address-family ipv4</td>
</tr>
<tr>
<td>Example:</td>
<td>Configures address-family for IS-IS.</td>
</tr>
<tr>
<td></td>
<td>This example shows the address-family for ipv4:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-sr-ms-map)# address-family ipv4</td>
</tr>
<tr>
<td></td>
<td>This example shows the address-family for ipv6:</td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-sr-ms-map)# address-family ipv6</td>
<td>Adds SID-mapping entries in the active local mapping policy. In the configured example:</td>
</tr>
<tr>
<td></td>
<td>• Prefix 10.1.1.1/32 is assigned prefix-SID 10, prefix 10.1.1.2/32 is assigned prefix-SID 11, ..., prefix 10.1.1.199/32 is assigned prefix-SID 200</td>
</tr>
<tr>
<td></td>
<td>• Prefix 20.1.0.0/16 is assigned prefix-SID 400, prefix 20.2.0.0/16 is assigned prefix-SID 401, ..., and so on.</td>
</tr>
</tbody>
</table>

**Step 6**

```
ip-address/prefix-length  first-SID-value  range  range
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)#
10.1.1.1/32 10 range 200
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)#
20.1.0.0/16 400 range 300
```

**Step 7**

```
commit
```

**Step 8**

```
show segment-routing mapping-server prefix-sid-map [ipv4 | ipv6] [detail]
```

**Example:**

```
 RP/0/RSP0/CPU0:router# show segment-routing mapping-server prefix-sid-map ipv4
Prefix          SID Index  Range
Flags
20.1.1.0/24     400         300
10.1.1.1/32     10          200

Number of mapping entries: 2
```

```
 RP/0/RSP0/CPU0:router# show segment-routing mapping-server prefix-sid-map ipv4 detail
Prefix          SID Index  Range         Last Prefix:          Last SID Index:  Flags:
20.1.1.0/24     400         300         20.2.44.0/24         699         Flags:
10.1.1.1/32     10          200         10.1.1.200/32         209         Flags:

Number of mapping entries: 2
```

**What to do next**

Enable the advertisement of the local SID-mapping policy in the IGP.
Enable Mapping Advertisement

In addition to configuring the static mapping policy, you must enable the advertisement of the mappings in the IGP.

Perform these steps to enable the IGP to advertise the locally configured prefix-SID mapping.

Configure Mapping Advertisement for IS-IS

SUMMARY STEPS

1. router isis instance-id
2. address-family { ipv4 | ipv6 } [ unicast ]
3. segment-routing prefix-sid-map advertise-local
4. commit
5. show isis database verbose

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>router isis instance-id</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<td> </td>
<td>• You can change the level of routing to be performed by a particular routing instance by using the is-type router configuration command.</td>
</tr>
<tr>
<td> </td>
<td>router isis 1</td>
</tr>
</tbody>
</table>

| Step 2 | address-family { ipv4 | ipv6 } [ unicast ] |
| **Example:** | Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |
| &nbsp; | The following is an example for ipv4 address family: |
| &nbsp; | router isis af unicast |

| Step 3 | segment-routing prefix-sid-map advertise-local |
| **Example:** | Configures IS-IS to advertise locally configured prefix-SID mappings. |
| &nbsp; | router isis af sf advertise-local |

| Step 4 | commit |

| Step 5 | show isis database verbose |
| **Example:** | Displays IS-IS prefix-SID mapping advertisement and TLV. |
| &nbsp; | show isis database verbose |
| &nbsp; | <...removed...> |
Configuration Advertisement for OSPF

**SUMMARY STEPS**

1. `router ospf process-name`
2. `segment-routing prefix-sid-map advertise-local`
3. `commit`
4. `show ospf database opaque-area`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> router ospf <em>process-name</em></td>
<td>Enables OSPF routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config)# router ospf 1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> segment-routing prefix-sid-map advertise-local</td>
<td>Configures OSPF to advertise locally configured prefix-SID mappings.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-ospf)# segment-routing prefix-sid-map advertise-local</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> commit</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> show ospf database opaque-area</td>
<td>Displays OSPF prefix-SID mapping advertisement and TLV.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router# show ospf database opaque-area</td>
<td></td>
</tr>
</tbody>
</table>

<...removed...> Extended Prefix Range TLV: Length: 24
AF: 0
Prefix: 10.1.1.1/32
Range Size: 200
Flags: 0x0

SID sub-TLV: Length: 8
Enable Mapping Client

By default, mapping client functionality is enabled.

You can disable the mapping client functionality by using the `segment-routing prefix-sid-map receive disable` command.

You can re-enable the mapping client functionality by using the `segment-routing prefix-sid-map receive` command.

The following example shows how to enable the mapping client for IS-IS:

```
RP/0/RSP0/CPU0:router(config)# router isis 1
RP/0/RSP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-isis-af)# segment-routing prefix-sid-map receive
```

The following example shows how to enable the mapping client for OSPF:

```
RP/0/RSP0/CPU0:router(config)# router ospf 1
RP/0/RSP0/CPU0:router(config-ospf)# segment-routing prefix-sid-map receive
```
Using Segment Routing Traffic Matrix

This module provides information about the Segment Routing Traffic Matrix (SR-TM) and the Traffic Collector process, and describes how to configure the TM border and the Traffic Collector and to display traffic information.

- Segment Routing Traffic Matrix, on page 93
- Traffic Collector Process, on page 93
- Configuring Traffic Collector, on page 94
- Displaying Traffic Information, on page 95

Segment Routing Traffic Matrix

A network’s traffic matrix is a description, measure, or estimation of the aggregated traffic flows that enter, traverse, and leave a network.

The Segment Routing Traffic Matrix (SR-TM) is designed to help users understand traffic patterns on a router. The Traffic Matrix border divides the network into two parts: internal (interfaces that are inside the border) and external (interfaces that are outside the border). By default, all interfaces are internal. You can configure an interface as external.

Traffic Collector Process

The Traffic Collector collects packet and byte statistics from router components such as prefix counters, tunnel counters, and the TM counter, which increments when traffic that comes from an external interface to the network is destined for a segment routing prefix-SID. The Traffic Collector keeps histories of the statistics and makes them persistent across process restarts, failovers, and ISSU. Histories are retained for a configurable length of time.

Pcounters

A Pcounter is a packet and byte pair of counters. There is one Pcounter per tunnel. There are two Pcounters per prefix-SID:

- Base Pcounter – any packet that is switched on the prefix-SID forwarding information base (FIB) entry
- TM Pcounter – any packet from an external interface and switched on the prefix-SID FIB entry
The Traffic Collector periodically collects the Base Pcounters and TM Pcounters of all prefix-SIDs, and the Pcounters of all tunnel interfaces.

For each Pcounter, the Traffic Collector calculates the number of packets and bytes that have been forwarded during the last interval. The Traffic Collector keeps a history of the per-interval statistics for each of the Pcounters. Each entry in the history contains:

- The start and end time of the interval
- The number of packets forwarded during the interval
- The number of bytes forwarded during the interval

## Configuring Traffic Collector

Perform these tasks to configure the traffic collector.

### SUMMARY STEPS

1. configure
2. traffic-collector
3. statistics collection-interval value
4. statistics history-size value
5. statistics history-timeout value
6. interface type l3-interface-address
7. commit

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 traffic-collector</td>
<td>Enables traffic collector and places the router in traffic collector configuration mode.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RP0/CPU0:router(config)# traffic-collector
```

<table>
<thead>
<tr>
<th>Step 3 statistics collection-interval value</th>
<th>(Optional) Sets the frequency that the traffic collector collects and posts data, in minutes. Valid values are 1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, and 60. The default interval is 1.</th>
</tr>
</thead>
</table>

**Example:**

```
RP/0/RP0/CPU0:router(config-tc)# statistics collection-interval 5
```

<table>
<thead>
<tr>
<th>Step 4 statistics history-size value</th>
<th>(Optional) Specifies the number of entries kept in the history database. Valid values are from 1 to 10. The default is 5.</th>
</tr>
</thead>
</table>

**Example:**

```
```
### Command or Action

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP/0/RP0/CPU0:router(config-tc)# statistics history-size 10</td>
<td>Note</td>
</tr>
<tr>
<td>RP/0/RP0/CPU0:router(config-tc)# statistics history-timeout 24</td>
<td>(Optional) When a prefix SID or a tunnel-te interface is deleted, the history-timeout sets the length of time, in hours, that the prefix SID and tunnel statistics are retained in the history before they are removed. The minimum is one hour; the maximum is 720 hours. The default is 48.</td>
</tr>
<tr>
<td>RP/0/RP0/CPU0:router(config-tc)# interface TenGigE 0/1/0/3</td>
<td>Identifies interfaces that handle external traffic. Only L3 interfaces are supported for external traffic.</td>
</tr>
</tbody>
</table>

### Displaying Traffic Information

The following show commands display information about the interfaces and tunnels:

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP/0/RP0/CPU0:router# show traffic-collector external-interface</td>
<td>For detailed information about the command syntax for the following <code>show</code> commands, see the Segment Routing Command Reference Guide.</td>
</tr>
<tr>
<td>Interface Status</td>
<td>---------</td>
</tr>
<tr>
<td>Te0/1/0/3 Enabled</td>
<td>---------</td>
</tr>
<tr>
<td>Te0/1/0/4 Enabled</td>
<td>---------</td>
</tr>
</tbody>
</table>

• Display the configured external interfaces:

```plaintext
RP/0/RP0/CPU0:router# show traffic-collector external-interface
Interface               Status
---------------------------------
Te0/1/0/3                Enabled
Te0/1/0/4                Enabled
```

• Display the counter history database for a prefix-SID:

```plaintext
RP/0/RP0/CPU0:router# show traffic-collector ipv4 counters prefix 1.1.1.10/32 detail
Prefix: 1.1.1.10/32   Label: 16010   State: Active
Base:
  Average over the last 5 collection intervals:
    Packet rate: 9496937 pps, Byte rate: 9363979882 Bps
```
History of counters:

23:01 - 23:02: Packets 9379529, Bytes: 9248215594
23:00 - 23:01: Packets 9687124, Bytes: 9551504264
22:59 - 23:00: Packets 9539200, Bytes: 9405651200

**TM Counters:**
Average over the last 5 collection intervals:
Packet rate: 9528754 pps, Byte rate: 9357236821 Bps

History of counters:

23:01 - 23:02: Packets 9400815, Bytes: 9231600330
23:00 - 23:01: Packets 9699455, Bytes: 9524864810
22:59 - 23:00: Packets 9579889, Bytes: 9407450998

This output shows the average Pcounter (packets, bytes), the Pcounter history, and the collection interval for the specified prefix-SID.

- Display the counter history database for a tunnel:

```
RP/0/RSP0/CPU0:router# show traffic-collector counters tunnels tunnel-te 1 detail
Tunnel: tt1 State: Active
Average over the last 5 collection intervals:
Packet rate: 9694434 pps, Byte rate: 9597489858 Bps

History of counters:

23:14 - 23:15: Packets 9870522, Bytes: 9771816780
23:13 - 23:14: Packets 9553048, Bytes: 9457517520
23:12 - 23:13: Packets 9647265, Bytes: 9550792350
23:11 - 23:12: Packets 9756654, Bytes: 9659087460
23:10 - 23:11: Packets 9694434, Bytes: 9548235180
```

This output shows the average Pcounter (packets, bytes), the Pcounter history, and the collection interval for the tunnel.
Using Segment Routing OAM

Segment Routing Operations, Administration, and Maintenance (OAM) helps service providers to monitor label-switched paths (LSPs) and quickly isolate forwarding problems to assist with fault detection and troubleshooting in the network. The Segment Routing OAM feature provides support for BGP prefix SID, IGP prefix SID and Nil-FEC (forwarding equivalence classes) LSP Ping and Traceroute functionality.

- MPLS Ping and Traceroute for BGP and IGP Prefix-SID, on page 97
- Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID, on page 98
- MPLS LSP Ping and Traceroute Nil FEC Target, on page 100
- Examples: LSP Ping and Traceroute for Nil_FEC Target, on page 100
- Segment Routing Ping, on page 101
- Segment Routing Traceroute, on page 104
- Segment Routing Policy Nil-FEC Ping and Traceroute, on page 106

MPLS Ping and Traceroute for BGP and IGP Prefix-SID

MPLS Ping and Traceroute operations for Prefix SID are supported for various BGP and IGP scenarios, for example:

- Within an IS-IS level or OSPF area
- Across IS-IS levels or OSPF areas
- Route redistribution from IS-IS to OSPF and from OSPF to IS-IS
- Anycast Prefix SID
- Combinations of BGP and LDP signaled LSPs

The MPLS LSP Ping feature is used to check the connectivity between ingress Label Switch Routers (LSRs) 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. 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.

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 MPLS LSP Tree Trace (traceroute multipath) operation is also supported for BGP and IGP Prefix SID. MPLS LSP Tree Trace provides the means to discover all possible equal-cost multipath (ECMP) routing paths of an LSP to reach a destination Prefix SID. It uses multipath data encoded in echo request packets to query for the load-balancing information that may allow the originator to exercise each ECMP. When the packet TTL expires at the responding node, the node returns the list of downstream paths, as well as the multipath information that can lead the operator to exercise each path in the MPLS echo reply. This operation is performed repeatedly for each hop of each path with increasing TTL values until all ECMP are discovered and validated.

MPLS echo request packets carry Target FEC Stack sub-TLVs. The Target FEC sub-TLVs are used by the responder for FEC validation. The BGP and IGP IPv4 prefix sub-TLV has been added to the Target FEC Stack sub-TLV. The IGP IPv4 prefix sub-TLV contains the prefix SID, the prefix length, and the protocol (IS-IS or OSPF). The BGP IPv4 prefix sub-TLV contains the prefix SID and the prefix length.

### Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID

These examples use the following topology:

![Topology Diagram](image)

**MPLS Ping for Prefix-SID**

```
RP/0/RSP0/CPU0# ping mpls ipv4 1.1.1.4/32
Thu Dec 17 01:01:42.301 PST
Sending 5, 100-byte MPLS Echos to 1.1.1.4, timeout is 2 seconds, send interval is 0 msec:
Type escape sequence to abort.

!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/2/3 ms
```
**MPLS Traceroute for Prefix-SID**

RP/0/RSP0/CPU0:router-arizona# traceroute mpls ipv4 1.1.1.4/32
Thu Dec 17 14:45:05.563 PST


Type escape sequence to abort.

0 12.12.12.1 MRU 4470 [Labels: 16004 Exp: 0]
L 1 12.12.12.2 MRU 4470 [Labels: 16004 Exp: 0] 3 ms
L 2 23.23.23.3 MRU 4470 [Labels: implicit-null Exp: 0] 3 ms
! 3 34.34.34.4 11 ms

**MPLS Tree Trace for Prefix-SID**

RP/0/RSP0/CPU0:router-arizona# traceroute mpls multipath ipv4 1.1.1.4/32
Thu Dec 17 14:55:46.549 PST

Starting LSP Path Discovery for 1.1.1.4/32


Type escape sequence to abort.

LL!
Path 0 found,
output interface TenGigE0/0/0/0 nexthop 12.12.12.2 source 12.12.12.1 destination 127.0.0.0
L!
Path 1 found,
output interface TenGigE0/0/0/0 nexthop 12.12.12.2 source 12.12.12.1 destination 127.0.0.2 LL!
Path 2 found,
output interface TenGigE0/0/0/1 nexthop 15.15.15.5 source 15.15.15.1 destination 127.0.0.1 L!
Path 3 found,
output interface TenGigE0/0/0/1 nexthop 15.15.15.5 source 15.15.15.1 destination 127.0.0.0
Paths (found/broken/unexplored) (4/0/0)
Echo Request (sent/fail) (10/0)
Echo Reply (received/timeout) (10/0)
Total Time Elapsed 53 ms
MPLS LSP Ping and Traceroute Nil FEC Target

The Nil-FEC LSP ping and traceroute operations are extensions of regular MPLS ping and traceroute.

Nil-FEC LSP Ping/Traceroute functionality supports segment routing and MPLS Static. It also acts as an additional diagnostic tool for all other LSP types. This feature allows operators to provide the ability to freely test any label stack by allowing them to specify the following:

- label stack
- outgoing interface
- nexthop address

In the case of segment routing, each segment nodal label and adjacency label along the routing path is put into the label stack of an echo request message from the initiator Label Switch Router (LSR); MPLS data plane forwards this packet to the label stack target, and the label stack target sends the echo message back.

The following table shows the syntax for the ping and traceroute commands.

Table 3: LSP Ping and Traceroute Nil FEC Commands

<table>
<thead>
<tr>
<th>Command Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>ping mpls nil-fec labels {label[,label]} [output {interface tx-interface}]</td>
</tr>
<tr>
<td>[nexthop nexthop-ip-addr]</td>
</tr>
<tr>
<td>traceroute mpls nil-fec labels {label[,label]} [output {interface tx-interface}</td>
</tr>
<tr>
<td>[nexthop nexthop-ip-addr]</td>
</tr>
</tbody>
</table>

Examples: LSP Ping and Traceroute for Nil_FEC Target

These examples use the following topology:

Node loopback IP address: 172.18.1.3 172.18.1.4 172.18.1.5 172.18.1.7
Node label: 16004 16005 16007
Nodes: Arizona ---- Utah ------- Wyoming ---- Texas
Interface: GigabitEthernet0/2/0/1 GigabitEthernet0/2/0/1
Interface IP address: 10.1.1.3 10.1.1.4

RP/0/RSP0/CPU0:router-utah# show mpls forwarding

<table>
<thead>
<tr>
<th>Local Label</th>
<th>Label</th>
<th>Prefix</th>
<th>Outgoing Prefix</th>
<th>Outgoing Interface</th>
<th>Next Hop</th>
<th>Bytes</th>
<th>Switched</th>
</tr>
</thead>
<tbody>
<tr>
<td>16004</td>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/1</td>
<td>10.1.1.4</td>
<td>16004</td>
<td>1392</td>
<td></td>
</tr>
<tr>
<td>16005</td>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/2</td>
<td>10.1.2.2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16007</td>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/0</td>
<td>10.1.1.4</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16007</td>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/1</td>
<td>10.1.2.2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16007</td>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/0</td>
<td>10.1.1.4</td>
<td>4752</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16007</td>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/1</td>
<td>10.1.2.2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24000</td>
<td>Pop</td>
<td>SR Adj (idx 0)</td>
<td>G10/2/0/0</td>
<td>10.1.1.4</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24001</td>
<td>Pop</td>
<td>SR Adj (idx 2)</td>
<td>G10/2/0/0</td>
<td>10.1.1.4</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Ping Nil FEC Target

RP/0/RSP0/CPU0:router-arizona# ping mpls nil-fec labels 16005,16007 output interface GigabitEthernet 0/2/0/1 nexthop 10.1.1.4 repeat 1
Sending 1, 72-byte MPLS Echos with Nil FEC labels 16005,16007, timeout is 2 seconds, send interval is 0 msec:


Type escape sequence to abort.
!
Success rate is 100 percent (1/1), round-trip min/avg/max = 1/1/1 ms
Total Time Elapsed 0 ms

Traceroute Nil FEC Target

RP/0/RSP0/CPU0:router-arizona# traceroute mpls nil-fec labels 16005,16007 output interface GigabitEthernet 0/2/0/1 nexthop 10.1.1.4
Tracing MPLS Label Switched Path with Nil FEC labels 16005,16007, timeout is 2 seconds


Type escape sequence to abort.
0 10.1.1.3 MRU 1500 [Labels: 16005/16007/explicit-null Exp: 0/0/0]
L 1 10.1.1.4 MRU 1500 [Labels: implicit-null/16007/explicit-null Exp: 0/0/0] 1 ms
L 2 10.1.1.5 MRU 1500 [Labels: implicit-null/explicit-null Exp: 0/0] 1 ms
! 3 10.1.1.7 1 ms

Segment Routing Ping

The MPLS LSP ping feature is used to check the connectivity between ingress and egress of 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. Segment routing ping is an extension of the MPLS LSP ping to perform the connectivity verification on the segment routing control plane.

**Note**

Segment routing ping can only be used when the originating device is running segment routing.

You can initiate the segment routing ping operation only when Segment Routing control plane is available at the originator, even if it is not preferred. This allows you to validate the SR path before directing traffic over the path. Segment Routing ping can use either generic FEC type or SR control-plane FEC type (SR-OSPF, SR-ISIS). In mixed networks, where some devices are running MPLS control plane (for example, LDP) or do not understand SR FEC, generic FEC type allows the device to successfully process and respond to the echo request. By default, generic FEC type is used in the target FEC stack of segment routing ping echo request. Generic FEC is not coupled to a particular control plane; it allows path verification when the advertising protocol is unknown or might change during the path of the echo request. If you need to specify the target FEC, you can select the FEC type as OSPF, IS-IS, or BGP. This ensures that only devices that are running segment routing control plane, and can therefore understand the segment routing IGP FEC, respond to the echo request.

**Configuration Examples**

These examples show how to use segment routing ping to test the connectivity of a segment routing control plane. In the first example, FEC type is not specified. You can also specify the FEC type as shown in the other examples.

```plaintext
RP/0/RSP0/CPU0:router# ping sr-mpls 10.1.1.2/32
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 = 1/2/5 ms
```

```plaintext
RP/0/RSP0/CPU0:router# ping sr-mpls 10.1.1.2/32 fec-type generic
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 = 1/1/2 ms

RP/0/RSP0/CPU0:router# ping sr-mpls 10.1.1.2/32 fec-type igp ospf

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 = 1/1/2 ms

RP/0/RSP0/CPU0:router# ping sr-mpls 10.1.1.2/32 fec-type igp isis

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 = 1/1/2 ms

RP/0/RSP0/CPU0:router# ping sr-mpls 10.1.1.2/32 fec-type bgp

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 = 1/1/2 ms
Segment Routing Traceroute

The MPLS LSP traceroute 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. Segment routing traceroute feature extends the MPLS LSP traceroute functionality to segment routing networks.

Similar to segment routing ping, you can initiate the segment routing traceroute operation only when Segment Routing control plane is available at the originator, even if it is not preferred. Segment Routing traceroute can use either generic FEC type or SR control-plane FEC type (SR-OSPF, SR-ISIS). By default, generic FEC type is used in the target FEC stack of segment routing traceroute echo request. If you need to specify the target FEC, you can select the FEC type as OSPF, IS-IS, or BGP. This ensures that only devices that are running segment routing control plane, and can therefore understand the segment routing IGP FEC, respond to the echo request.

The existence of load balancing at routers in an MPLS network provides alternate paths for carrying MPLS traffic to a target router. The multipath segment routing traceroute feature provides a means to discover all possible paths of an LSP between the ingress and egress routers.

Configuration Examples

These examples show how to use segment routing traceroute to trace the LSP for a specified IPv4 prefix SID address. In the first example, FEC type is not specified. You can also specify the FEC type as shown in the other examples.

```
RP/0/RSP0/CPU0:router# traceroute sr-mpls 10.1.1.2/32
Tracing MPLS Label Switched Path to 10.1.1.2/32, timeout is 2 seconds
Type escape sequence to abort.
   0 10.12.12.1 MRU 1500 [Labels: implicit-null Exp: 0]
   1 10.12.12.2 3 ms

RP/0/RSP0/CPU0:router# traceroute sr-mpls 10.1.1.2/32 fec-type generic
Tracing MPLS Label Switched Path to 10.1.1.2/32, timeout is 2 seconds
```
This example shows how to use multipath traceroute to discover all the possible paths for an IPv4 prefix SID.
Segment Routing Policy Nil-FEC Ping and Traceroute

Segment routing OAM supports Nil-FEC LSP ping and traceroute operations to verify the connectivity for segment routing MPLS data plane. For the existing Nil-FEC ping and traceroute commands, you need to specify the entire outgoing label stack, outgoing interface, as well as the next hop. SR policy Nil-FEC ping and SR policy Nil-FEC traceroute enhancements extend the data plane validation functionality of installed SR policies through Nil-FEC ping and traceroute commands while simplifying the operational process. Instead of specifying the entire outgoing label-stack, interface, and next-hop, you can use the policy name or the policy binding-SID label value to initiate Nil-FEC ping and traceroute operations for the SR policies. Specification of outgoing interface and next-hop is also not required for policy Nil-FEC OAM operations.

Restrictions and Usage Guidelines

The following restrictions and guidelines apply for this feature:

• You cannot select a specific candidate path for SR policy Nil-FEC ping and traceroute.

• You cannot use SR policy Nil-FEC ping or traceroute for non-selected candidate paths.

Examples: SR Policy Nil-FEC Ping

These examples show how to use SR policy Nil-FEC ping for a SR policy. The first example refers the SR policy-name while the second example refers the BSID.

RP/0/0/CPU0:router# ping sr-mpls nil-fec policy name POLICY1
Thu Feb 22 06:56:50.006 PST
Sending 5, 100-byte MPLS Echos with Nil FEC for SR-TE Policy POLICY1, 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 = 1/5/22 ms

RP/0/0/CP00:router# ping sr-mpls nil-fec policy-binding-sid 100001
Thu Dec 17 12:41:02.381 EST
Sending 5, 100-byte MPLS Echos with Nil FEC with labels [16002,16003],
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 = 2/3/3 ms

Examples: SR Policy Nil-FEC Traceroute

These examples show how to use SR policy Nil-FEC traceroute for a SR policy. The first example refers the
SR policy-name while the second example refers the binding SID (BSID).

RP/0/0/CP00:router# traceroute sr-mpls nil-fec policy name POLICY1
Thu Feb 22 06:57:03.637 PST
Tracing MPLS Label Switched Path with Nil FEC for SR-TE Policy POLICY1, 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 11.11.11.1 MRU 1500 [Labels: 16003/explicit-null Exp: 0/0]
L 1 11.11.11.2 MRU 1500 [Labels: implicit-null/explicit-null Exp: 0/0] 4 ms
L 1 14.14.14.3 2 ms

RP/0/0/CP00:router# traceroute sr-mpls nil-fec binding-sid
Thu Dec 17 12:41:02.381 EST
Sending 5, 100-byte MPLS Echos with Nil FEC with labels [16002,16003],
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.
0 111.11.11.1 MRU 1500 [Labels: 16003/explicit-null Exp: 0/0]
L 1 111.11.11.2 MRU 1500 [Labels: implicit-null/explicit-null Exp: 0/0] 4 ms
L 1 14.14.14.3 2 ms

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.5.x