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)

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

PREFACE

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

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 over IPv6 (SRv6) 7
Segment Routing over IPv6 Base 7
Configuring SRv6 IS-IS 11
SRv6-Based IPv4 L3VPN 12
SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode 16
SRv6 Services for L3VPN Active-Standby Redundancy using Port-Active Mode: Operation 17
Configure SRv6 Services L3VPN Active-Standby Redundancy using Port-Active Mode 17
Configuration Example 17
Running Configuration 18
Verification 18

CHAPTER 4
Configure Segment Routing Global Block and Segment Routing Local Block 23
About the Segment Routing Global Block 23
Color-Only Steering 66
Address-Family Agnostic Steering 67
Flexible Name-based Policy Constraints 68
  Configuring Flexible Name-Based Policy Constraints 68
  Affinity Support for Anycast SIDs 70
LDP over Segment Routing Policy 73
Static Route Traffic-Steering using SRTE Policy 76
BGP SR-TE 77
  Configure Explicit BGP SR-TE 77
  Setting CO Flag 80
Configure Interface TE Metrics 80
Configure Interface Link Admin Groups and SR-TE Affinity Maps 81
On-Demand SR Policy – SR On-Demand Next-Hop 82
  Configuring SR-ODN: Examples 86
  Configuring SR-ODN for EVPN-VPWS: Use Case 93
Configure the Head-End Router as PCEP PCC 113
Using Binding Segments 116
  Stitching SR-TE Polices Using Binding SID: Example 117

 CHAPTER 9
Enabling Segment Routing Flexible Algorithm 121
  Prerequisites for Flexible Algorithm 121
Building Blocks of Segment Routing Flexible Algorithm 121
  Flexible Algorithm Definition 121
  Flexible Algorithm Support Advertisement 122
  Flexible Algorithm Definition Advertisement 122
  Flexible Algorithm Prefix-SID Advertisement 122
  Calculation of Flexible Algorithm Path 122
  Installation of Forwarding Entries for Flexible Algorithm Paths 123
Configuring Flexible Algorithm 123
  Example: Configuring IS-IS Flexible Algorithm 124
  Example: Traffic Steering to Flexible Algorithm Paths 125
  BGP Routes on PE – Color Based Steering 125

 CHAPTER 10
Configure Segment Routing Path Computation Element 129
CHAPTER 15

Using Segment Routing OAM 161
  MPLS Ping and Traceroute for BGP and IGP Prefix-SID 161
  Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID 162
  MPLS LSP Ping and Traceroute Nil FEC Target 164
  Examples: LSP Ping and Traceroute for Nil_FEC Target 164
  Segment Routing Ping 165
  Segment Routing Traceroute 168
  Segment Routing Policy Nil-FEC Ping and Traceroute 170
  Segment Routing over IPv6 OAM 172
Preface

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

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

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>April 2019</td>
<td>Initial release of this document</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

The table below summarizes the new and modified Segment Routing features in IOS XR Release 6.6.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>Static Route Traffic-Steering using SRTE Policy</td>
<td>This feature is introduced.</td>
<td>Release 6.6.2</td>
<td>Static Route Traffic-Steering using SRTE Policy, on page 76</td>
</tr>
<tr>
<td>ACL Support for PCEP Connection</td>
<td>This feature is introduced.</td>
<td>Release 6.6.2</td>
<td>ACL Support for PCEP Connection, on page 135</td>
</tr>
<tr>
<td>SRv6 Base</td>
<td>This feature is introduced.</td>
<td>Release 6.6.1</td>
<td>Segment Routing over IPv6 Base, on page 7</td>
</tr>
<tr>
<td>SRv6 IS-IS</td>
<td>This feature is introduced.</td>
<td>Release 6.6.1</td>
<td>Configuring SRv6 IS-IS, on page 11</td>
</tr>
<tr>
<td>SR Flexible Algorithm for IS-IS</td>
<td>This feature is introduced.</td>
<td>Release 6.6.1</td>
<td>Enabling Segment Routing Flexible Algorithm</td>
</tr>
<tr>
<td>SRv6-based IPv4 L3VPN</td>
<td>This feature is introduced.</td>
<td>Release 6.6.1</td>
<td>SRv6-Based IPv4 L3VPN, on page 12</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Introduced/Changed in Release</td>
<td>Where Documented</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode</td>
<td>This feature is introduced.</td>
<td>Release 6.6.1</td>
<td>SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode, on page 16</td>
</tr>
<tr>
<td>SRV6 OAM</td>
<td>This feature is introduced.</td>
<td>Release 6.6.1</td>
<td>Segment Routing over IPv6 OAM, on page 172</td>
</tr>
</tbody>
</table>
About Segment Routing

Note
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 policy 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 policy.

**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 policies 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 over IPv6 (SRv6)

Segment Routing for IPv6 (SRv6) is the implementation of Segment Routing over the IPv6 dataplane.

- Segment Routing over IPv6 Base, on page 7
- Configuring SRv6 IS-IS, on page 11
- SRv6-Based IPv4 L3VPN, on page 12
- SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode, on page 16

Segment Routing over IPv6 Base

Segment Routing (SR) can be applied on both MPLS and IPv6 data planes. This feature extends Segment Routing support with IPv6 data plane. In an SR-MPLS enabled network, an MPLS label is used as the Segment Identifier (SID) and the source router chooses a path to the destination and encodes the path in the packet header as a stack of labels. However, in a Segment Routing over IPv6 (SRv6) network, an IPv6 address serves as the SID. The source router encodes the path to destination as an ordered list of segments (list of IPv6 addresses) in the IPv6 packet. To encode an ordered list of IPv6 addresses in an IPv6 packet, a new routing header which is an extension header is used. This new header for SRv6 is called Segment Routing Header (SRH). In an SRv6 enabled network, the active segment is indicated by the destination address of the packet, and the next segment is indicated by a pointer in the SRH.

The following list explains the fields in SRH:

- Next header—Identifies the type of header immediately following the SRH.
- Hdr Ext Len (header extension length)—The length of the SRH in 8-octet units, not including the first 8 octets.
- Segments left—Specifies the number of route segments remaining. That means, the number of explicitly listed intermediate nodes still to be visited before reaching the final destination.
- Last Entry—Contains the index (zero based) of the last element of the segment list.
- Flags—Contains 8 bits of flags.
- Tag—Tag a packet as part of a class or group of packets like packets sharing the same set of properties.
- Segment list—128-bit IPv6 addresses representing the nth segment in the segment list. The segment list encoding starts from the last segment of the SR policy (path). That means the first element of the segment list (Segment list [0]) contains the last segment of the SR policy, the second element contains the penultimate segment of the SR policy and so on.
Each node along the SRv6 packet path has a different functionality:

- **Source node**—A node that can generate an IPv6 packet with an SRH (an SRv6 packet), or an ingress node that can impose an SRH on an IPv6 packet.

- **Transit node**—A node along the path of the SRv6 packet (IPv6 packet and SRH). The transit node does not inspect the SRH. The destination address of the IPv6 packet does not correspond to the transit node.

- **End point node**—A node in the SRv6 domain where the SRv6 segment is terminated. The destination address of the IPv6 packet with an SRH corresponds to the end point node. The segment endpoint node executes the function bound to the SID.

### Table 1: Example of a Segment Routing Header

<table>
<thead>
<tr>
<th>Next Header</th>
<th>Hdr Ext Len</th>
<th>Routing Type</th>
<th>Segments Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Entry</td>
<td>Flags</td>
<td>Segment List[0]</td>
<td>Tag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(128-bit IPv6 address)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Segment List[n]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(128-bit IPv6 address)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optional Type Length Value objects (variable)</td>
<td></td>
</tr>
</tbody>
</table>

In SRv6, a SID represents a 128-bit value, consisting of the following three parts:

- **Locator**: This is the first part of the SID with most significant bits and represents an address of a specific SRv6 node.

- **Function**: This is the portion of the SID that is local to the owner node and designates a specific SRv6 function (network instruction) that is executed locally on a particular node, specified by the locator bits.

- **Args**: This field is optional and represents optional arguments to the function.

The locator part can be further divided into two parts:

- **SID Block**: This field is the SRv6 network designator and is a fixed or known address space for an SRv6 domain. This is the most significant bit (MSB) portion of a locator subnet.

- **Node Id**: This field is the node designator in an SRv6 network and is the least significant bit (LSB) portion of a locator subnet.

### Configuring SRv6 Base

To enable SRv6 globally, you should first configure a locator with its prefix. The IS-IS protocol announces the locator prefix in IPv6 network and SRv6 applications (like ISIS, BGP) use it to allocate SIDs.

The following restrictions and usage guidelines apply while configuring SRv6:

- All routers in the SRv6 domain should have the same SID block (network designator) in their locator.

- The locator length should be 64-bits long.
• The SID block portion (MSBs) cannot exceed 40 bits. If this value is less than 40 bits, user should use a pattern of zeros as a filler.

• The Node Id portion (LSBs) cannot exceed 24 bits.

• Only a single locator is supported and that will be the default locator for SID allocation.

Enabling SRv6 with Locator

This example shows how to globally enable SRv6 and configure locator.

```
RP/0/0/CPU0:Router(config)# segment-routing srv6
RP/0/0/CPU0:Router(config-srv6)# locators
RP/0/0/CPU0:Router(config-srv6-locators)# locator Loc1
RP/0/0/CPU0:Router(config-srv6-locator)# prefix 2001:db8:0:a2::/64
```

Optional: Configuring Encapsulation Parameters

This example shows how to configure encapsulation parameters when configuring SRv6. These optional parameters include:

• Source Address of outer encapsulating IPv6 header: The default source address for encapsulation is one of the loopback addresses,

• Hop Limit of outer encapsulating IPv6 header: The default value for hop-limit is 255.

```
RP/0/0/CPU0:Router(config)# segment-routing srv6
RP/0/0/CPU0:Router(config-srv6)# encapsulation source-address 1::1
RP/0/0/CPU0:Router(config-srv6)# hop-limit 60
```

Optional: Enabling Syslog Logging for Locator Status Changes

This example shows how to enable the logging of locator status.

```
RP/0/0/CPU0:Router(config)# segment-routing srv6
RP/0/0/CPU0:Router(config-srv6)# logging locator status
```

Verifying SRv6 Manager

This example shows how to verify the overall SRv6 state from SRv6 Manager point of view. The output displays parameters in use, summary information, and platform specific capabilities.

```
RP/0/0/CPU0:Router# show segment-routing srv6 manager
Parameters:
SRv6 Enabled: Yes
Encapsulation:
  Source Address:
    Configured: 1::1
    Default: 5::5
  Hop-Limit: Default
Summary:
  Number of Locators: 1 (1 operational)
  Number of SIDs: 4 (0 stale)
  Max SIDs: 64000
```
OOR:
Thresholds: Green 3200, Warning 1920
Status: Resource Available (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: Yes
  Microloop-Avoidance: No
End Functions:
  End (PSP)
  End.X (PSP)
  End.DX4
  End.DT4
Transit Functions:
  T
  T.Insert.Red
  T.Encaps.Red
Security rules:
  SEC-1
  SEC-2
  SEC-3
  SEC-4
Counters:
  CNT-1
  CNT-3
Signaled Parameters:
  Max-SL  : 4
  Max-End-Pop-SRH : 4
  Max-T-Insert : 4
  Max-T-Encap : 5
  Max-End-D : 5
Max SIDs: 64000
SID Holdtime: 30 mins

Verifying SRv6 Locator

This example shows how to verify the locator configuration and its operational status.

```
RP/0/0/CP00:Router# show segment-routing srv6 locator myLoc1 detail
Name   ID  Prefix       Status
-------------------------------
myLoc1* 5  2001:db8:0:a2::/64 Up
(*): is-default
Interface:
  Name: srv6-myLoc1
  IFH : 0x00000170
  IPv6 address: 2001:db8:0:a2::/64
  Chkpt Obj ID: 0x2fc8
  Created: Apr 25 06:21:57.077 (00:03:37 ago)
```

Verifying SRv6 SIDs

This example shows how to verify the allocation of SRv6 local SIDs off locator(s).

```
RP/0/0/CP00:Router# show segment-routing srv6 locator myLoc1 sid
SID     Function  Context    Owner
------------------------ -------- ------------------
2001:db8:0:a2::1:     End (PSP)  'default':1 sidmgr
```
The following example shows how to display detail information regarding an allocated SRv6 local SID.

```
RP/0/0/CPU0:Router# show segment-routing srv6 locator myLoc1 sid 2001:db8:0:a2:40:: detail
```

```
SID Function Context Owner
-------------------------- ----------- ------------------------------ ------------------
----- --
2001:db8:0:a2:40:: End.DT4 'VRF1' bgp-100
InUse Y

SID context: { table-id=0xe0000011 ('VRF1':IPv4/Unicast) }
Locator: myLoc1'
Allocation type: Dynamic
Created: Feb 1 14:04:02.901 (3d00h ago)
```

Similarly, you can display SID information across locators by using the `show segment-routing sid` command.

**show Commands**

You can use the following `show` commands to verify the SRv6 global and locator configuration:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>show segment-routing srv6 manager</code></td>
<td>Displays the summary information from SRv6 manager, including platform capabilities.</td>
</tr>
<tr>
<td><code>show segment-routing srv6 locator locator-name [detail]</code></td>
<td>Displays the SRv6 locator information on the router.</td>
</tr>
<tr>
<td><code>show segment-routing srv6 locator locator-name sid [sid-ipv6-address [detail]]</code></td>
<td>Displays the information regarding SRv6 local SID(s) allocated from a given locator.</td>
</tr>
<tr>
<td>`show segment-routing srv6 sid [sid-ipv6-address</td>
<td>all</td>
</tr>
<tr>
<td><code>show route ipv6 local-srv6</code></td>
<td>Displays all SRv6 local-SID prefixes in IPv6 RIB.</td>
</tr>
</tbody>
</table>

### Configuring SRv6 IS-IS

Intermediate System-to-Intermediate System (IS-IS) protocol already supports segment routing with MPLS dataplane (SR-MPLS). This feature enables extensions in IS-IS to support Segment Routing with IPv6 data plane (SRv6). The extensions include advertising the SRv6 capabilities of nodes and node and adjacency segments as SRv6 SIDs.

SRv6 IS-IS performs the following functionalities:

1. Interacts with SID Manager to learn local locator prefixes and announces the locator prefixes in the IGP domain.
2. Learns remote locator prefixes from other ISIS neighbor routers and installs the learned remote locator IPv6 prefix in RIB or FIB.

3. Allocate or learn prefix SID and adjacency SIDs, create local SID entries, and advertise them in the IGP domain.

Restrictions and Usage Guidelines

The following restrictions and usage guidelines apply for SRv6 IS-IS in Cisco IOS XR release 6.6.1.

- Segment routing for traffic engineering (SR-TE) is not supported.
- Segment routing microloop avoidance is not supported.
- An ISIS address-family can support either SR-MPLS or SRv6, but both at the same time is not supported.

Configuring SRv6 IS-IS

To configure SRv6 IS-IS, you should enable SRv6 under the IS-IS IPv6 address-family. The following example shows how to configure SRv6 IS-IS.

```
RP/0/0/CPU0:Router(config)# router isis srv6
RP/0/0/CPU0:Router(config-isis)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-af)# segment-routing srv6
RP/0/0/CPU0:Router(config-isis-srv6)# locator locator1
RP/0/0/CPU0:Router(config-isis-srv6-loc)# exit
```

SRv6-Based IPv4 L3VPN

The SRv6-based IPv4 L3VPN feature enables deployment of IPv4 L3VPN over a SRv6 data plane. Traditionally, it was done over an MPLS-based system. SRv6-based L3VPN uses SRv6 Segment IDs (SIDs) for service segments instead of labels. SRv6-based L3VPN functionality interconnects multiple sites to resemble a private network service over public infrastructure. To use this feature, you must configure SRv6-base.

For this feature, BGP allocates an SRv6 SID from the locator space, configured under SRv6-base and VPNv4 address family. For more information on this, refer Segment Routing over IPv6 Base, on page 7. The BGP SID can be allocated in the following ways:

- Per-VRF mode that provides End.DT4 support. End.DT4 represents the Endpoint with decapsulation and IPv4 table lookup.
- Per-CE mode that provides End.DX4 cross connect support. End.DX4 represents the Endpoint with decapsulation and IPv4 cross-connect.

BGP encodes the SRv6 SID in the prefix-SID attribute of the IPv4 L3VPN Network Layer Reachability Information (NLRI) and advertises it to IPv6 peering over an SRv6 network. The Ingress PE (provider edge) router encapsulates the VRF IPv4 traffic with the SRv6 VPN SID and sends it over the SRv6 network.

Restrictions and Usage Guidelines

- MPLS based L3VPN inter-operability is not supported on a router that is configured for SRv6-based L3VPN.
• Only IPv4 L3VPN is supported, and IPv6 L3VPN is not supported.
• Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
• BGP, OSPF, Static are supported as PE-CE protocol.

Configuring SRv6 based IPv4 L3VPN

To enable SRv6-based L3VPN, you need to configure SRv6 under BGP and SID allocation mode. The following example shows how to configure SRv6-based L3VPN:

/*Configure BGP and specify locator name*/
RP/0/0/CPU0:Router(config)# router bgp 100
RP/0/0/CPU0:Router(config-bgp)# bgp router-id 10.6.6.6
RP/0/0/CPU0:Router(config-bgp)# address-family vpnv4 unicast
RP/0/0/CPU0:Router(config-bgp-af)# segment-routing srv6
RP/0/0/CPU0:Router(config-bgp-af-srv6)# locator my-locator
RP/0/0/CPU0:Router(config-bgp-af-srv6)# exit

/*Configure a VRF with per-vrf label allocation mode*/
RP/0/0/CPU0:Router(config-bgp-af)# vrf vrf1
RP/0/0/CPU0:Router(config-bgp-vrf)# rd 106:1
RP/0/0/CPU0:Router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-bgp-vrf)# segment-routing srv6
RP/0/0/CPU0:Router(config-bgp-vrf-af-srv6)# alloc mode per-vrf
RP/0/0/CPU0:Router(config-bgp-vrf-af-srv6)# exit
RP/0/0/CPU0:Router(config-bgp-vrf-af)# exit
RP/0/0/CPU0:Router(config-bgp-vrf)# neighbor 10.1.2.2
RP/0/0/CPU0:Router(config-bgp-vrf-nbr)# remote-as 100
RP/0/0/CPU0:Router(config-bgp-vrf-af)# address-family ipv4 unicast

/*Configure a VRF with per-ce label allocation mode*/
RP/0/0/CPU0:Router(config-bgp-af)# vrf vrf2
RP/0/0/CPU0:Router(config-bgp-vrf)# rd 106:2
RP/0/0/CPU0:Router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-bgp-vrf)# segment-routing srv6
RP/0/0/CPU0:Router(config-bgp-vrf-af-srv6)# alloc mode per-ce
RP/0/0/CPU0:Router(config-bgp-vrf-af-srv6)# exit
RP/0/0/CPU0:Router(config-bgp-vrf-af)# exit
RP/0/0/CPU0:Router(config-bgp-vrf)# neighbor 10.1.2.2
RP/0/0/CPU0:Router(config-bgp-vrf-nbr)# remote-as 100
RP/0/0/CPU0:Router(config-bgp-vrf-af)# address-family ipv4 unicast

Verification

The following example shows how to verify the SRv6 based L3VPN configuration using the show segment-routing srv6 sid command.

In this example, End.X represents Endpoint function with Layer-3 cross-connect, End.DT4 represents Endpoint with decapsulation and IPv4 table lookup, and End.DX4 represents Endpoint with decapsulation and IPv4 cross-connect.

RP/0/0/CPU0:Router# show segment-routing srv6 sid
*** Locator: 'my_locator' ***
SID     State  RW Function Context Owner
-------- ----------- ------------ ---------------
cafe:0:0:66:1::   End (PSP) 'my_locator':1 sidmgr
InUse   Y

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x
The following example shows how to verify the SRv6 based L3VPN configuration using the `show segment-routing srv6 SID-prefix detail` command.

```
RP/0/RP0/CPU0:Router#sh segment-routing srv6 sid cafe:0:0:66:43:: detail
Sun Dec 9 16:52:54.015 EST
*** Locator: 'my_locator' ***
SID Function Context Owner
State RW-------------------------- ----------- ------------------------------ ------------------
----- --
cafe:0:0:66:44:: End.DT4 'VRF1' bgp-100
InUse Y
SID context: { table-id=0xe0000001 ('VRF1':IPv4/Unicast) }
Locator: 'my_locator'
Allocation type: Dynamic
Created: Dec 8 16:34:32.506 (1d00h ago)
```

The following example shows how to verify the SRv6 based L3VPN configuration using the `show bgp vpnv4 un rd route-distinguisher prefix` command on Egress PE.

```
RP/0/RP0/CPU0:SRv6-Hub6#sh bgp vpnv4 un rd route-distinguisher prefix
date
Wed Nov 21 16:08:44.765 EST
BGP routing table entry for 10.15.0.0/30, Route Distinguisher: 106:1
Versions:
  Process bRIB/RIB SendTblVer
  Speaker 2282449 2282449
SRv6-VPN SID: cafe:0:0:66:44::/128
Last Modified: Nov 21 15:50:34.235 for 00:18:10
Paths: (2 available, best #1)
  Advertised to peers (in unique update groups):
  2::2
  Path #1: Received by speaker 0
```
Advertised to peers (in unique update groups):
  2:12
  200
    10.1.2.2 from 10.1.2.2 (10.7.0.1)
      Origin IGP, localpref 200, valid, internal, best, group-best, import-candidate
      Received Path ID 0, Local Path ID 1, version 2276228
      Extended community: RT:201:1
    Path #2: Received by speaker 0
    Not advertised to any peer
  200
    10.2.2.2 from 10.2.2.2 (10.20.1.2)
      Origin IGP, localpref 100, valid, internal
      Received Path ID 0, Local Path ID 0, version 0
      Extended community: RT:201:1

The following example shows how to verify the SRv6 based L3VPN configuration using the `show bgp vpnv4 un rd route-distinguisher prefix` command on Ingress PE.

```
RP/0/RP0/CPU0:SRv6-LF1#sh bgp vpnv4 un rd 106:1 10.15.0.0/30
Wed Nov 21 16:11:45.538 EST
BGP routing table entry for 10.15.0.0/30, Route Distinguisher: 106:1
Versions:
  Process: bRIB/RIB  SendTblVer
  Speaker:  2286222  2286222
Last Modified: Nov 21 15:47:26.288 for 00:24:19
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
    Not advertised to any peer
    200, (received & used)
      6::6 (metric 24) from 2::2 (6.6.6.6)
      Received Label 3
      Origin IGP, localpref 200, valid, internal, best, group-best, import-candidate,
      not-in-vrf
      Received Path ID 1, Local Path ID 1, version 2286222
      Extended community: RT:201:1
      Originator: 6.6.6.6, Cluster list: 2.2.2.2
      SRv6-VPN-SID: T1-cafe:0:0:66:44:: [total 1]
```

The following example shows how to verify the SRv6 based L3VPN configuration using the `show route vrf vrf-name prefix detail` command.

```
RP/0/RP0/CPU0:Router#sh route vrf VRF1 10.15.0.0/30 detail
Wed Nov 21 16:35:17.775 EST
Routing entry for 10.15.0.0/30
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Installed Nov 21 16:35:14.107 for 00:00:03
Routing Descriptor Blocks
  6::6, from 2::2
    Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
    Route metric is 0
    Label: None
    Tunnel ID: None
    Binding Label: None
    Extended communities count: 0
    Source RD attributes: 0x0000:106:1
    NHID:0x0 (Ref:0)
    SRv6 Transit Type: T.Encaps.Red
    SRv6 SID-list { cafe:0:0:66:44:: }
    MPLS eid:0x13806000000001
    Route version is 0xd (13)
    No local label
```
The following example shows how to verify the SRv6 based L3VPN configuration for per-ce allocation mode using the `show bgp vrf vrf nexthop-set` command.

```
RP/0/RP0/CPU0:Router#show bgp vrf VRF2 nexthop-set
Wed Nov 21 15:52:17.464 EST
Resilient per-CE nexthop set, ID 3
Number of nexthops 1, Label 0, Flags 0x2200
SRv6-VPN SID: cafe:0:0:66:46::/128
Nexthops:
10.1.2.2
Reference count 1, Resilient per-CE nexthop set, ID 4
Number of nexthops 2, Label 0, Flags 0x2100
SRv6-VPN SID: cafe:0:0:66:47::/128
Nexthops:
10.1.2.2
10.2.2.2
Reference count 2,
```

The following example shows how to verify the SRv6 based L3VPN configuration using the `show cef vrf vrf-name prefix detail location line-card` command.

```
RP/0/RP0/CPU0:Router#sh cef vrf VRF1 10.15.0.0/30 detail location 0/0/cpu0
Wed Nov 21 16:37:06.894 EST
151.1.0.0/30, version 3038384, SRv6 Transit, internal 0x5000001 0x0 (ptr 0x9ae6474c) [1], 0x0 (0x0), 0x0 (0x8c11b238)
Updated Nov 21 16:35:14.109
Prefix Len 30, traffic index 0, precedence n/a, priority 3
gateway array (0x8cd85190) reference count 1014, flags 0x2010, source rib (7), 0 backups
[1 type 3 flags 0x40441 (0x8a529798) ext 0x0 (0x0)]
LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
gateway array update type-time 1 Nov 21 14:47:26.816
LDI Update time Nov 21 14:52:53.073
Level 1 - Load distribution: 0
[0] via cafe:0:0:66::/128, recursive
via cafe:0:0:66::/128, 7 dependencies, recursive [flags 0x6000]
path-idx 0 NHID 0x0 [0x8acb53cc 0x0]
next hop VRF - 'default', table - 0xe0800000
next hop cafe:0:0:66::/128 via cafe:0:0:66::/64
SRv6 T.Encaps.Red SID-list {cafe:0:0:66:44::}
Load distribution: 0 (reecount 1)
Hash OK Interface Address
0 Y Bundle-Ether1201 fe80::2
```

## SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode

The Segment Routing IPv6 (SRv6) Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode feature provides all-active per-port load balancing for multihoming. The forwarding of traffic is
determined based on a specific interface rather than per-flow across multiple Provider Edge routers. This feature enables efficient load-balancing and provides faster convergence. In an active-standby scenario, the active PE router is detected using designated forwarder (DF) election by modulo calculation and the interface of the standby PE router brought down. For Modulo calculation, byte 10 of the Ethernet Segment Identifier (ESI) is used.

Restrictions

• This feature can only be configured for bundle interfaces.
• When an EVPN Ethernet Segment (ES) is configured with port-active load-balancing mode, you cannot configure ACs of that bundle on bridge-domains with a configured EVPN instance (EVI). EVPN Layer 2 bridging service is not compatible with port-active load-balancing.

SRv6 Services for L3VPN Active-Standby Redundancy using Port-Active Mode: Operation

Under port-active operational mode, EVPN Ethernet Segment (ES) routes are exchanged across BGP for the routers servicing the multihomed ES. Each PE router then builds an ordered list of the IP addresses of all PEs connected to the ES, including itself, and assigns itself an ordinal for its position in the list. The ordinals are used with the modulo calculation to determine which PE will be the Designated Forwarder (DF) for a given ES. All non-DF PEs will take the respective bundles out of service.

In the case of link or port failure, the active DF PE withdraws its ES route. This re-triggers DF election for all PEs that service the ES and a new PE is elected as DF.

Configure SRv6 Services L3VPN Active-Standby Redundancy using Port-Active Mode

This section describes how you can configure SRv6 services L3VPN active-standby redundancy using port-active mode under an Ethernet Segment (ES).

Configuration Example

/* Configure Ethernet Link Bundles */
Router# configure
Router(config)# interface Bundle-Ether10
Router(config-if)# ipv4 address 10.0.0.2 255.255.255.0
Router(config-if)# ipv6 address 2001:DB8::1
Router(config-if)# lacp period short
Router(config-if)# mac-address 1.2.3
Router(config-if)# bundle wait-while 0
Router(config-if)# exit
Router(config)# interface GigabitEthernet 0/2/0/5
Router(config-if)# bundle id 14 mode active
Router(config-if)# commit

/* Configure load balancing. */
Router# configure
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether10
Router(config-evpn-ac)# ethernet-segment
Configure Segment Routing over IPv6 (SRv6)

Running Configuration

```conf
Router(config-evpn-ac-es)# identifier type 0 11.11.11.11.11.11.11.14
Router(config-evpn-ac-es)# load-balancing-mode port-active
Router(config-evpn-ac-es)# commit

/* Configure address family session in BGP. */
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 192.168.0.2
Router(config-bgp)# address-family 12vpn evpn
Router(config-bgp)# neighbor 192.168.0.3
Router(config-bgp-nbr)# remote-as 200
Router(config-bgp-nbr)# update-source Loopback 0
Router(config-bgp-nbr)# address-family 12vpn evpn
Router(config-bgp-nbr)# commit
```

Verification

Verify the SRv6 services L3VPN active-standby redundancy using port-active mode configuration.

```bash
/* Verify ethernet-segment details on active DF router */
Router# show evpn ethernet-segment interface Bundle-Ether14 detail
Ethernet Segment Id Interface Nexthops
----------------------------------------------- ------------------------------------
0011.1111.1111.1111.1111.1114 BE14 192.168.0.2
ES to BGP Gates : Ready
ES to L2FIB Gates : Ready
Main port :
    Interface name : Bundle-Ether14
```

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x
Interface MAC : 0001.0002.0003
IfHandle : 0x000041d0
State : Up
Redundancy : Not Defined
ESI type : 0
Value : 11.1111.1111.1111.1114
ES Import RT : 1111.1111.1111 (from ESI)
Source MAC : 0000.0000.0000 (N/A)
Topology :
  Operational : MH
  Configured : Port-Active
Service Carving : Auto-selection
Multicast : Disabled
Peering Details :
  192.168.0.2 [MOD:P:00]
  192.168.0.3 [MOD:P:00]

Service Carving Results:
  Forwards : 0
  Permanent : 0
  Elected : 0
  Not Elected : 0
MAC Flushing mode : STP-TCN
Peering timer : 3 sec [not running]
Recovery timer : 30 sec [not running]
Carving timer : 0 sec [not running]
Local SHG label : None
Remote SHG labels : 0

/* Verify bundle Ethernet configuration on active DF router */
Router# show bundle bundle-ether 14
Bundle-Ether14
  Status: Up
  Local links <active/standby/configured>: 1 / 0 / 1
  Local bandwidth <effective/available>: 1000000 (1000000) kbps
  MAC address (source): 0001.0002.0003 (Configured)
  Inter-chassis link: No
  Minimum active links / bandwidth: 1 / 1 kbps
  Maximum active links: 64
  Wait while timer: Off
  Load balancing:
    Link order signaling: Not configured
    Hash type: Default
    Locality threshold: None
  LACP: Operational
  Flap suppression timer: Off
  Cisco extensions: Disabled
  Non-revertive: Disabled
  mLACP: Not configured
  IPv4 BFD: Not configured
  IPv6 BFD: Not configured

  Port        Device     State     Port ID     B/W, kbps
  ------------- ----------- ----------- -------------- ----------
  Gi0/2/0/5   Local      Active     0x8000, 0x0003  1000000
  Link is Active

/* Verify ethernet-segment details on standby DF router */
Router# show evpn ethernet-segment interface bundle-ether 10 detail
Router# show evpn ethernet-segment interface Bundle-Ether24 detail
Ethernet Segment Id | Interface | Nexthops
------------------------ | ---------------------------------- | ----------------------
0011.1111.1111.1111.1114 BE24 | 192.168.0.2 | 192.168.0.3

ES to BGP Gates: Ready
ES to L2FIB Gates: Ready
Main port:
  Interface name: Bundle-Ether24
  Interface MAC: 0001.0002.0003
  IfHandle: 0x000041b0
  State: Standby
  Redundancy: Not Defined
  ESI type: 0
  Value: 11.1111.1111.1111.1114
  ES Import RT: 1111.1111.1111 (from ESI)
  Source MAC: 0000.0000.0000 (N/A)
Topology:
  Operational: MH
  Configured: Port-Active
Service Carving: Auto-selection
Multicast: Disabled
Peering Details:
  192.168.0.2 [MOD:P:00]
  192.168.0.3 [MOD:P:00]

Service Carving Results:
  Forwarders: 0
  Permanent: 0
  Elected: 0
  Not Elected: 0
MAC Flushing mode: STP-TCN
Peering timer: 3 sec [not running]
Recovery timer: 30 sec [not running]
Carving timer: 0 sec [not running]
Local SHG label: None
Remote SHG labels: 0

/* Verify bundle configuration on standby DF router */
Router# show bundle bundle-ether 24

Bundle-Ether24
Status: LACP OOS (out of service)
Local links <active/standby/configured>: 0 / 1 / 1
Local bandwidth <effective/available>: 0 (0) kbps
MAC address (source): 0001.0002.0003 (Configured)
Inter-chassis link: No
Minimum active links / bandwidth: 1 / 1 kbps
Maximum active links: 64
Wait while timer: Off
Load balancing:
  Link order signaling: Not configured
  Hash type: Default
  Locality threshold: None
  LACP: Operational
  Flap suppression timer: Off
  Cisco extensions: Disabled
  Non-revertive: Disabled
mLACP: Not configured
IPv4 BFD: Not configured
IPv6 BFD: Not configured

Port | Device | State | Port ID | B/W, kbps
-----------------------------------------------

Gi0/0/0/4 Local Standby 0x8000, 0x0002 1000000
Link is in standby due to bundle out of service state
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 23
- About the Segment Routing Local Block, on page 24
- Setup a Non-Default Segment Routing Global Block Range, on page 25
- Setup a Non-Default Segment Routing Local Block Range, on page 26

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.

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>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>configure</td>
<td></td>
</tr>
<tr>
<td>Step 2</td>
<td>`[router {isis instance-id</td>
<td>ospf process_name} ]`</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config)# router isis 1</code></td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td><code>segment-routing global-block starting_value ending_value</code></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>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-isis)# segment-routing global-block 18000 19999</code></td>
<td></td>
</tr>
<tr>
<td>Step 4</td>
<td>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=Gi0/0/0/0, 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><strong>Step 1</strong> <code>configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>segment-routing local-block starting_value ending_value</code></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><code>RP/0/RSP0/CPU0:router(config)# segment-routing local-block 30000 30999</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>commit</code></td>
<td></td>
</tr>
</tbody>
</table>

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 5

Configure Segment Routing for IS-IS Protocol


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

Note

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 29
- Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 32
- Configuring an Adjacency SID, on page 34
- Configuring Bandwidth-Based Local UCMP, on page 40
- IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability, on page 41
- IS-IS Multi-Domain Prefix SID and Domain Stitching: Example, on page 44

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:  
<p>|
| | Enables RSVP traffic engineering functionality. |</p>
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis-af)# mpls traffic-eng level-2-only</td>
<td></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><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-isis-af)# mpls traffic-eng router-id Loopback0</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> router-id loopback loopback interface used for prefix-sid</td>
<td>Configures router ID for each address-family (ipv4/ipv6).</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/(config-isis-af)#router-id loopback0</td>
<td></td>
</tr>
<tr>
<td><strong>Step 8</strong> segment-routing mpls</td>
<td>Segment routing is enabled by the following actions:</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-isis-af)# segment-routing mpls</td>
<td>• MPLS forwarding is enabled on all interfaces where IS-IS is active.</td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td>• The prefix-SIDs locally configured are advertised.</td>
</tr>
<tr>
<td><strong>Step 9</strong> exit</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>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-isis-af)# exit</td>
<td></td>
</tr>
<tr>
<td><strong>Step 10</strong> mpls traffic-eng</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# mpls traffic-eng</td>
<td></td>
</tr>
<tr>
<td><strong>Step 11</strong> commit</td>
<td></td>
</tr>
</tbody>
</table>

**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 policies. 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 policies. 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-spfs | algorithm algorithm-number ] { 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>router isis instance-id</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# router isis 1</td>
</tr>
<tr>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<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>
<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>
</tbody>
</table>
### Purpose

#### Command or Action

<table>
<thead>
<tr>
<th>Step 3</th>
<th>interface Loopback instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td></td>
</tr>
</tbody>
</table>

#### Purpose

Specifies the loopback interface and instance.

| Step 4 | address-family { ipv4 | ipv6 } [ unicast ]  |
|--------|----------------------------------|
| Example: | | RP/0/RSP0/CPU0:router(config-isis-if)# address-familyipv4 unicast |

#### Purpose

Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

| Step 5 | prefix-sid [strict-spf | algorithm algorithm-number] |
|--------|----------------------------|
| [index SID-index | absolute SID-value] [n-flag-clear] |
| [explicit-null ]  |
| Example: | | RP/0/RSP0/CPU0:router(config-isis-if-af)# prefix-sid index 1001 |

#### Purpose

Configures the prefix-SID index or absolute value for the interface.

Specify **strict-spf** to configure the prefix-SID to use the SPF path instead of the SR-TE policy.

Specify **algorithm algorithm-number** to configure SR Flexible Algorithm. See Enabling Segment Routing Flexible Algorithm, on page 121.

Specify **index SID-index** for each node to create a prefix SID based on the lower boundary of the SRGB + the index.

Specify **absolute SID-value** for each node to create a specific prefix SID within the SRGB.

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-clear keyword. IS-IS does not set the n flag in the prefix-SID sub Type Length Value (TLV).

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.

**Note** 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.

### Verify the prefix-SID configuration:

<table>
<thead>
<tr>
<th>Step 6</th>
<th>commit</th>
</tr>
</thead>
</table>

Verify the prefix-SID configuration:

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

<table>
<thead>
<tr>
<th>LSPID</th>
<th>LSP Seq Num</th>
<th>LSP Checksum</th>
<th>LSP Holdtime</th>
<th>ATT/P/OL</th>
</tr>
</thead>
<tbody>
<tr>
<td>router.00-00</td>
<td>* 0x0000039b</td>
<td>0x0c2f</td>
<td>1079</td>
<td>0/0/0</td>
</tr>
</tbody>
</table>
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.
### Table 2: 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>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> 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>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td><code>router isis 1</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> interface type interface-path-id</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-isis)#</td>
<td></td>
</tr>
<tr>
<td><code>interface GigabitEthernet0/0/0/7</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> point-to-point</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></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>RP/0/RSP0/CPU0:router(config-isis-if)# point-to-point</td>
<td>Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.</td>
</tr>
</tbody>
</table>

**Step 5**

| address-family { ipv4 | ipv6 } [ unicast ] 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 ] 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 | Configures the Adj-SID index or absolute value for the interface. Specify index adj-SID-index for each link to create an Adj-SID based on the lower boundary of the SRLB + the index. Specify absolute adj-SID-value for each link to create a specific Adj-SID within the SRLB. Specify if the Adj-SID is protected. For each primary path, 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. |

**Step 7**

| commit | 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

<table>
<thead>
<tr>
<th>Local Label</th>
<th>Outgoing Prefix</th>
<th>Outgoing Interface</th>
<th>Next Hop</th>
<th>Bytes Switched</th>
</tr>
</thead>
<tbody>
<tr>
<td>15010 Pop</td>
<td>SRLB (idx 10)</td>
<td>Gi0/0/0/3</td>
<td>10.0.3.3</td>
<td>0</td>
</tr>
<tr>
<td>Pop</td>
<td>SRLB (idx 10)</td>
<td>Gi0/0/0/7</td>
<td>10.1.0.5</td>
<td>0</td>
</tr>
<tr>
<td>16004</td>
<td>SRLB (idx 10)</td>
<td>Gi0/0/0/7</td>
<td>10.1.0.5</td>
<td>0 (!)</td>
</tr>
</tbody>
</table>
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 ]`
### Configure Segment Routing for IS-IS Protocol

**Configuring a Layer 2 Adjacency SID**

6. `l2-adjacency sid {index adj-SID-index | absolute adj-SID-value} [next-hop {ipv4_address | ipv6_address}]`

7. `commit`

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>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>configure</code></td>
<td>Enters segment routing configuration mode.</td>
</tr>
<tr>
<td>2</td>
<td><code>segment-routing</code></td>
<td>Enters adjacency SID configuration mode.</td>
</tr>
<tr>
<td></td>
<td><code>adjacency-sid</code></td>
<td>Specifies the interface and enters interface configuration mode.</td>
</tr>
<tr>
<td></td>
<td><code>interface type interface-path-id</code></td>
<td>Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.</td>
</tr>
<tr>
<td></td>
<td>`address-family {ipv4</td>
<td>ipv6} [unicast]`</td>
</tr>
</tbody>
</table>

**Steps 6, 7, 8, and 9**

**Example:**

- `Router(config-sr-adj-intf-af)# l2-adjacency sid index 5 absolute 15015 next-hop 10.1.1.4`

---

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.
<table>
<thead>
<tr>
<th>Step 7</th>
<th>commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 8</td>
<td>end</td>
</tr>
</tbody>
</table>
| **Step 9** | `router isis instance-id`  
**Example:**  
`Router(config)# router isis isp` |
| **Step 10** | `address-family { ipv4 | ipv6 } [ unicast ]`  
**Example:**  
`Router(config-isis)# address-family ipv4 unicast` |
| **Step 11** | `segment-routing bundle-member-adj-sid`  
**Example:**  
`Router(config-isis-af)# segment-routing bundle-member-adj-sid` |

**Purpose**

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.

**Enable IS-IS routing for the specified routing instance, and places the router in router configuration mode.**

**Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.**

**Programs the dynamic Layer 2 Adj-SIDs, and advertises both manual and dynamic Layer 2 Adj-SIDs.**

**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:**

```bash
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
        12-adjacency-sid absolute 15001
```

**Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x**

39
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 |
| | 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 is-type router configuration command. |
| **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 |
| | Specifies the IPv4 or IPv6 address family, and enters IS-IS address family configuration mode. |
| **Step 4** | apply-weight ecmp-only bandwidth  
**Example:**  
RP/0/RSP0/CPU0:router(config-isis-af)# apply-weight ecmp-only bandwidth |
| | Enables UCMP functionality locally between ECMP paths based on the bandwidth of the local links. |
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 3: Source Router Sub-TLV Format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N (Node Flag)</td>
<td>For prefixes that are propagated from another level:</td>
</tr>
</tbody>
</table>

| 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 3: Source Router Sub-TLV Format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv4 Source Router ID</td>
<td>Type: 11</td>
</tr>
<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>

| IPv6 Source Router ID| Type: 12                                                                                                                                     |
|                      | Length: 16                                                                                                                                   |
|                      | Value: IPv6 Router ID of the source of the prefix advertisement                                                                           |
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 configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 interface Loopback instance</td>
<td>Specifies the loopback interface and instance.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>interface Loopback0</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:</td>
<td></td>
</tr>
<tr>
<td>isis prefix-attributes n-flag-clear</td>
<td></td>
</tr>
<tr>
<td>Step 4 commit</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 Segment Routing Path Computation Element (SR-PCE) (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

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.

Configure BGP Link-state (BGP-LS) on Node 13 and Node 14 to report their local domain to Node 10:
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 47
- Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 49

### 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> <code>router ospf process-name</code></td>
<td>Enables segment routing using the MPLS data plane on the routing process and all areas and interfaces in the routing process.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# <code>router ospf 1</code></td>
<td>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><strong>Step 3</strong> <code>segment-routing mpls</code></td>
<td>Enables IGP traffic engineering functionality.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-ospf)# <code>segment-routing mpls</code></td>
<td>Sets the traffic engineering loopback interface.</td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>area 0</code></td>
<td>Enters area configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-ospf)# <code>area 0</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> <code>mpls traffic-eng area</code></td>
<td>Enables IGP traffic engineering functionality.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-ospf-ar)# <code>mpls traffic-eng area 0</code></td>
<td>Sets the traffic engineering loopback interface.</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><code>mpls</code></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><code>segment-routing mpls</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 SR-TE head-end requires the TED to calculate and validate the path of the SR-TE policy.</td>
</tr>
<tr>
<td><code>exit</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 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

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

1. **interface Loopback** `interface-instance`
2. **prefix-sid** `[strict-spf]` `{index SID-index | absolute SID-value }` `[n-flag-clear]` `[explicit-null]`
3. **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>configure</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><code>router ospf</code> <code>process-name</code></td>
<td>Enables OSPF routing for the specified routing process, 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 ospf 1</code></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><code>area</code> <code>value</code></td>
<td>Enters area configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:router(config-ospf)# area 0</code></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>interface Loopback</code> <code>interface-instance</code></td>
<td>Specifies the loopback interface and instance.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:router(config-ospf-ar)# interface Loopback0 passive</code></td>
<td></td>
</tr>
</tbody>
</table>
| 5    | **prefix-sid** `[strict-spf]` `{index SID-index | absolute SID-value }` `[n-flag-clear]` `[explicit-null]` | Configures the prefix-SID index or absolute value for the interface.  
Specify **strict-spf** to configure the prefix-SID to use the SPF path instead of the SR-TE policy.  
Specify **index SID-index** for each node to create a prefix SID based on the lower boundary of the SRGB + the index.  
Specify **absolute SID-value** for each node to create a specific prefix SID within the SRGB.  
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.  
OSPF does not set the n flag in the prefix-SID sub Type Length Value (TLV).  
To disable penultimate-hop-popping (PHP) and add an explicit-Null label, enter the explicit-null keyword. OSPF sets the n flag in the prefix-SID sub TLV. |
|      | Example:          |         |
|      | `RP/0/RSP0/CPU0:router(config-ospf-ar)# prefix-sid index 1001` | |
|      | `RP/0/RSP0/CPU0:router(config-ospf-ar)# prefix-sid absolute 17001` | |
| 6    | **commit**        |         |

Verify the prefix-SID configuration:

```
RP/0/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 7

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.

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 53
- Configure BGP Prefix Segment Identifiers, on page 54
- Configure Segment Routing Egress Peer Engineering, on page 55
- Configure BGP Link-State, on page 56
- Example: Configuring SR-EPE and BGP-LS, on page 57
- Configure BGP Proxy Prefix SID, on page 59

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 BPG (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)# show bgp 1.1.1.3/32
```

BGP routing table entry for 1.1.1.3/32

Versions:
- Process: bRIB/RIB, SendTblVer: 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>&lt;br&gt;Example: <code>RP/0/RSP0/CPU0:router(config)# router bgp 1</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>Step 2</td>
<td><code>neighbor ip-address</code>&lt;br&gt;Example: <code>RP/0/RSP0/CPU0:router(config-bgp)# neighbor 192.168.1.3</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>Step 3</td>
<td><code>remote-as as-number</code>&lt;br&gt;Example:</td>
<td>Creates a neighbor and assigns a remote autonomous system number to it.</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 BGPLS data from remote peers. BGP does not download the received BGPLS data to any other component on the router.

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

- **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:

```bash
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, to applications that desire to build paths spanning or including these multiple domains.

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

```bash
RP/0/RSP0/CPU0:router# configure
```
To distribute OSPFv2 and OSPFv3 link-state data using BGP LS, use the `distribute bgp-ls` command in router configuration mode.

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# router ospf 100
RP/0/RSP0/CPU0:router(config-ospf)# distribute bgp-ls instance-id 32 throttle 10
```

**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**

![Topology Diagram]

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

*Example:*

```
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
```

**Step 2** Configure node C to advertise peer node SIDs to the controller using BGP-LS.
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 Type</th>
<th>Prefix or ID</th>
<th>Interface</th>
<th>Next Hop</th>
<th>Switched</th>
</tr>
</thead>
<tbody>
<tr>
<td>24002</td>
<td>Unlabelled</td>
<td>No ID</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>No ID</td>
<td>Te0/1/0/0</td>
<td>192.168.1.3</td>
<td>0</td>
</tr>
</tbody>
</table>
The output shows that node C installed peer node SIDs in the Forwarding Information Base (FIB).

## 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.

```plaintext
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.

```plaintext
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.

```plaintext
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
  Flags:
```
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
  Path #1: Received by speaker 0
        Advertised to peers (in unique update groups):
          201.1.1.1
Local
  20.0.101.1 from 20.0.101.1 (20.0.101.1) Received Label 61
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 192.68.64.1/32
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
  20.0.101.1, from 20.0.101.1, 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 0x7a, source rib (7), 0 backups
  2 type 5 flags 0x88401 (0x0722eb450) ext 0x0 (0x0)
Updated Oct 31 23:23:48.733
Prefix Len 32, traffic index 0, precedence n/a, priority 4
Extensions: context-label:16400
gateway array (0x71ae7e78) reference count 3, flags 0x7a, source rib (7), 0 backups
  [2 type 5 flags 0x88401 (0x0722eb450) ext 0x0 (0x0)]
LW-LDI[type=5, refc=3, ptr=0x711c11590, sh-ldi=0x0722eb450] gateway array update type-time 3 Oct 31 23:49:11.720
LDI Update time Oct 31 23:23:48.733
via 20.0.101.1/32, 0 dependencies, recursive, bgp-ext [flags 0x6020]
  path-idx 0 NHID 0x0 [0x7129s294 0x0]
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: 0xe0000000 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>&gt;11.1.1.1/32</td>
<td>1.1.1.1</td>
<td>3</td>
<td>16010</td>
</tr>
<tr>
<td>&gt; 2.1.1.1/32</td>
<td>0.0.0.0</td>
<td>nolabel</td>
<td>3</td>
</tr>
<tr>
<td>&gt; 192.68.64.1/32</td>
<td>20.0.101.1</td>
<td>2</td>
<td>16400</td>
</tr>
<tr>
<td>&gt; 192.68.64.2/32</td>
<td>20.0.101.1</td>
<td>2</td>
<td>16401</td>
</tr>
</tbody>
</table>
Configure BGP Proxy Prefix SID
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 63
- BGP SR-TE, on page 77
- Configure Interface TE Metrics, on page 80
- Configure Interface Link Admin Groups and SR-TE Affinity Maps, on page 81
- On-Demand SR Policy – SR On-Demand Next-Hop, on page 82
- Configure the Head-End Router as PCEP PCC, on page 113
- Using Binding Segments, on page 116

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 Segment Routing Path Computation Element (SR-PCE). For information on configuring SR-PCE, see Configure Segment Routing Path Computation Element, on page 129 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

/* 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
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).

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 1.1.1.23/32
Router(config-sr-te-policy)# autoroute include ipv4 1.1.1.21/32
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 80.

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 ::

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 Segment Routing Path Computation Element (SR-PCE). The binding SID (BSID) state notification for each policy contains an "ipv6_caps" flag that notifies SR-PCE clients (PCC) 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.

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 G10/0/0/0
Router(config-sr-te-if)# affinity
Router(config-sr-te-if-affinity)# blue

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x
Configure SR-TE Policies

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 32 or Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 49.

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:

```plaintext
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.
Candidate-paths:
  Preference 100:
  Constraints:
    Affinity:
      exclude-any: red
    Explicit: segment-list SIDLIST1 (active)
  Weight: 0, Metric Type: IGP
  16100 [Prefix-SID, 1.1.1.8]
  16004 [Prefix-SID, 4.4.4.4]

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.
Eventhough parallel link (23) is marked with red, it is still considered valid since anycast traffic flows only on the path to node 5.

LDP over Segment Routing Policy

The LDP over Segment Routing Policy feature enables an LDP-targeted adjacency over a Segment Routing (SR) policy between two routers. This feature extends the existing MPLS LDP address family neighbor configuration to specify an SR policy as the targeted end-point.

LDP over SR policy is supported for locally configured SR policies with IPv4 end-points.

For more information about MPLS LDP, see the "Implementing MPLS Label Distribution Protocol" chapter in the MPLS Configuration Guide.

Before you configure an LDP targeted adjacency over SR policy name, you need to create the SR policy under Segment Routing configuration. The SR policy interface names are created internally based on the color and endpoint of the policy. LDP is non-operational if SR policy name is unknown.

The following functionality applies:

1. Configure the SR policy – LDP receives the associated end-point address from the interface manager (IM) and stores it in the LDP interface database (IDB) for the configured SR policy.

2. Configure the SR policy name under LDP – LDP retrieves the stored end-point address from the IDB and uses it. Use the auto-generated SR policy name assigned by the router when creating an LDP targeted adjacency over an SR policy. Auto-generated SR policy names use the following naming convention: srte_c_color_val_ep_endpoint-address. For example, srte_c_1000_ep_1.1.1.2

Limitations for LDP over Segment Routing Policy

Configuration Example

/* Enter the SR-TE configuration mode and create the SR policy. This example corresponds to a local SR policy with an explicit path. */
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Configure SR-TE Policies

LDP over Segment Routing Policy

```
Router(config-sr-te)# segment-list sample-sid-list
Router(config-sr-te-sl)# index 10 address ipv4 1.1.1.7
Router(config-sr-te-sl)# index 20 address ipv4 1.1.1.2
Router(config-sr-te-sl)# exit
Router(config-sr-te)# policy sample_policy
Router(config-sr-te-policy)# color 1000 end-point ipv4 1.1.1.2
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# explicit segment-list sample-sid-list
Router(config-sr-te-pp-info)# end

;/* Configure LDP over an SR policy */
Router(config)# mpls ldp
Router(config-ldp)# address-family ipv4
Router(config-ldp-af)# neighbor sr-policy srte_c_1000_ep_1.1.1.2 targeted
Router(config-ldp-af)#
```

Note

Do one of the following to configure LDP discovery for targeted hellos:

- **Active targeted hellos (SR policy head end):**
  ```
  mpls ldp
  interface GigabitEthernet0/0/0/0
  !
  !
  ```

- **Passive targeted hellos (SR policy end-point):**
  ```
  mpls ldp
  address-family ipv4
  discovery targeted-hello accept
  !
  !
  ```

Running Configuration

```
segment-routing
traffic-eng
  segment-list sample-sid-list
    index 10 address ipv4 1.1.1.7
    index 20 address ipv4 1.1.1.2
  !
  policy sample_policy
  color 1000 end-point ipv4 1.1.1.2
  candidate-paths
  preference 100
  explicit segment-list sample-sid-list
  !
  !
  !

mpls ldp
  address-family ipv4
  neighbor sr-policy srte_c_1000_ep_1.1.1.2 targeted
discovery targeted-hello accept
```

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x
Verification

Router# show mpls ldp interface brief

<table>
<thead>
<tr>
<th>Interface</th>
<th>VRF Name</th>
<th>Config</th>
<th>Enabled</th>
<th>IGP-Auto-Cfg</th>
<th>TE-Mesh-Grp</th>
<th>cfg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Te0/3/0/0/3</td>
<td>default</td>
<td>Y</td>
<td>Y</td>
<td>0</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Te0/3/0/0/6</td>
<td>default</td>
<td>Y</td>
<td>Y</td>
<td>0</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Te0/3/0/0/7</td>
<td>default</td>
<td>Y</td>
<td>Y</td>
<td>0</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Te0/3/0/0/8</td>
<td>default</td>
<td>N</td>
<td>N</td>
<td>0</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Te0/3/0/0/9</td>
<td>default</td>
<td>N</td>
<td>N</td>
<td>0</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>srte_c_1000_</td>
<td>default</td>
<td>Y</td>
<td>Y</td>
<td>0</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

Router# show mpls ldp interface

Interface TenGigE0/3/0/0/3 (0xa000340)
VRF: 'default' (0x60000000)
Enabled via config: LDP interface

Interface TenGigE0/3/0/0/6 (0xa000400)
VRF: 'default' (0x60000000)
Enabled via config: LDP interface

Interface TenGigE0/3/0/0/7 (0xa000440)
VRF: 'default' (0x60000000)
Enabled via config: LDP interface

Interface TenGigE0/3/0/0/8 (0xa000480)
VRF: 'default' (0x60000000)
Disabled:

Interface TenGigE0/3/0/0/9 (0xa0004c0)
VRF: 'default' (0x60000000)
Disabled:

Interface srte_c_1000_ep_1.1.1.2 (0x520)
VRF: 'default' (0x60000000)
Enabled via config: LDP interface

Router# show segment-routing traffic-eng policy color 1000

SR-TE policy database
--------------------------------
Color: 1000, End-point: 1.1.1.2
Name: srte_c_1000_ep_1.1.1.2
Status:
  Admin: up  Operational: up for 00:02:00 (since Jul 2 22:39:06.663)
Candidate-paths:
  Preference: 100 (configuration) (active)
    Name: sample_policy
    Requested BSID: dynamic
    PCC info:
      Symbolic name: cfg_sample_policy_discr_100
      PLSP-ID: 17
    Explicit: segment-list sample-sid-list (valid)
      Weight: 1, Metric Type: TR
      16007 [Prefix-SID, 1.1.1.7]
      16002 [Prefix-SID, 1.1.1.2]
Attributes:
  Binding SID: 80011
  Forward Class: 0
  Steering BGP disabled: no
  IPv6 caps enable: yes
Static Route Traffic-Steering using SRTE Policy

In previous releases, you could only associate Segment Routing Label Switched Paths (SR-LSP) with a static route. The Static Route Traffic-Steering using SRTE Policy feature allows you to specify a Segment Routing (SR) policy as an interface type when configuring static routes for MPLS and IPv6 data planes.

For information on configuring static routes, see the "Implementing Static Routes" chapter in the Routing Configuration Guide for Cisco ASR 9000 Series Routers.

Configuration Example

Router(config)# router static
Router (config-static)# address-family ipv4 unicast

//configure administrative distance
Router (config-static-afi)# 1.1.1.1/32 sr-policy policy1 110

//Configure load metric
Router (config-static-afi)# 1.1.1.1/32 sr-policy policy1 metric 5

//Install the route in RIB regardless of reachability
Router (config-static-afi)# 1.1.1.1/32 sr-policy policy1 permanent

Running Configuration

configure
router static
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.

**Note**
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 SR-TE policy with specific characteristics and explicit paths. On the receiver side, an SR-TE policy 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 policy for each path.

**Configure Explicit BGP SR-TE**
Perform this task to configure explicit BGP SR-TE:

**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
### 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>configure</td>
<td></td>
</tr>
<tr>
<td>Step 2</td>
<td>extcommunity-set opaque name</td>
<td>Defines the color extended community-set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# extcommunity-set opaque color1</td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td>name</td>
<td>Defines the color extended community-set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-ext)# 1</td>
<td></td>
</tr>
<tr>
<td>Step 4</td>
<td>end-set</td>
<td>Ends the definition of the extended community-set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-ext)# end-set</td>
<td></td>
</tr>
<tr>
<td>Step 5</td>
<td>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></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# route-policy color</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-rpl)# if destination in (5.5.5.1/32) then</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-rpl-if)# set extcommunity color color1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-rpl-if)# endif</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-rpl)# end-policy</td>
<td></td>
</tr>
<tr>
<td>Step 6</td>
<td>end-policy</td>
<td>Ends the definition of a route policy and exits route policy configuration mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-rpl)# end-policy</td>
<td></td>
</tr>
<tr>
<td>Step 7</td>
<td>router bgp as-number</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></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# router bgp 1</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td><strong>Configure the local router with a specified router ID.</strong></td>
<td></td>
</tr>
<tr>
<td><code>bgp router-id ip-address</code></td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 10.10.0.2</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 9</strong></td>
<td><strong>Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.</strong></td>
<td></td>
</tr>
<tr>
<td>`address-family {ipv4</td>
<td>ipv6} sr-policy`</td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 sr-policy</code></td>
</tr>
<tr>
<td><strong>Step 10</strong></td>
<td><strong>Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.</strong></td>
<td></td>
</tr>
<tr>
<td><code>exit</code></td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.0.1</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 11</strong></td>
<td><strong>Creates a neighbor and assigns a remote autonomous system number to it.</strong></td>
<td></td>
</tr>
<tr>
<td><code>remote-as as-number</code></td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 1</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 12</strong></td>
<td><strong>Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.</strong></td>
<td></td>
</tr>
<tr>
<td>`address-family {ipv4</td>
<td>ipv6} unicast`</td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast</code></td>
</tr>
<tr>
<td><strong>Step 13</strong></td>
<td><strong>Applies the specified policy to IPv4 unicast routes.</strong></td>
<td></td>
</tr>
<tr>
<td>`route-policy route-policy-name {in</td>
<td>out}`</td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy color out</code></td>
</tr>
<tr>
<td><strong>Step 14</strong></td>
<td><strong>Sends extended community attributes to external Border Gateway Protocol (eBGP) neighbors.</strong></td>
<td></td>
</tr>
<tr>
<td><code>send-extended-community-ebgp</code></td>
<td><strong>Example:</strong>&lt;br&gt;<code>RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# send-extended-community-ebgp</code></td>
<td></td>
</tr>
</tbody>
</table>
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 SR-PCE 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

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

| co-flag 00 | 1. The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>.  
|           | 2. If a policy does not exist, then IGP path for the next-hop N is chosen.  |
| co-flag 01 | 1. The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>.  
|           | 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.  
|           | 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.  
|           | 4. If no match is found, then IGP path for the next-hop N is chosen.  |

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)#
```

Configure Interface TE Metrics

Use the `metric value` command in SR-TE interface submode to configure the TE metric for interfaces. The value range is from 0 to 2147483647.

```
Router# configure
Router(config)# segment-routing  
Router(config-sr)# traffic-eng
```
Configuring TE Metric: Example

The following configuration example shows how to set the TE metric for various interfaces:

```
segment-routing
  traffic-eng
    interface TenGigE0/0/0/0
      metric 100
    !
    interface TenGigE0/0/0/1
      metric 1000
    !
    interface TenGigE0/0/2/0
      metric 50
    !
end
```

Configure Interface Link Admin Groups and SR-TE Affinity Maps

Use the `affinity name NAME` command in SR-TE interface submode to assign affinity to interfaces. Configure this on routers with interfaces that have an associated admin group attribute.

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# interface type interface-path-id
Router(config-sr-if)# affinity
Router(config-sr-if-affinity)# name NAME
```

Use the `affinity-map name NAME bit-position bit-position` command in SR-TE sub-mode to define affinity maps. The `bit-position` range is from 0 to 255.

Configure affinity maps on the following routers:

- Routers with interfaces that have an associated admin group attribute.
- Routers that act as SR-TE head-ends for SR policies that include affinity constraints.

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# affinity-map
Router(config-sr-te-affinity-map)# name NAME bit-position bit-position
```

Configuring Link Admin Group: Example

The following example shows how to assign affinity to interfaces and to define affinity maps.

```
segment-routing
  traffic-eng
    interface TenGigE0/0/1/1
      affinity
        name CROSS
        name RED
    !
 Professor
interface TenGigE0/0/1/2
  affinity
    name RED
!*
interface TenGigE0/0/2/0
  affinity
    name BLUE
!*
affinity-map
  name RED bit-position 23
  name BLUE bit-position 24
  name CROSS bit-position 25
!*
end

Configuring Affinity Maps at SR-TE Head-End: Example

The following example shows how to define affinity maps at an SR-TE head-end router with an SR policy that includes affinity constraints.

segment-routing
traffic-eng
  policy foo
color 100 end-point ipv4 1.1.1.2
candidate-paths
  preference 100
dynamic
  metric
    type te
!*
  !*
constraints
  affinity
    exclude-any
    name RED
!*
  !*
  !*
affinity-map
  name RED bit-position 23
  name BLUE bit-position 24
!*

On-Demand SR Policy – SR On-Demand Next-Hop

Segment Routing On-Demand Next Hop (SR-ODN) allows a service head-end router to automatically instantiate an SR policy to a BGP next-hop when required (on-demand). Its key benefits include:

- **SLA-aware BGP service** – Provides per-destination steering behaviors where a prefix, a set of prefixes, or all prefixes from a service can be associated with a desired underlay SLA. The functionality applies equally to single-domain and multi-domain networks.
**Simplicity** – No prior SR Policy configuration needs to be configured and maintained. Instead, operator simply configures a small set of common intent-based optimization templates throughout the network.

**Scalability** – Device resources at the head-end router are used only when required, based on service or SLA connectivity needs.

The following example shows how SR-ODN works:

1. An egress PE (node H) advertises a BGP route for prefix T/t. This advertisement includes an SLA intent encoded with a BGP color extended community. In this example, the operator assigns color purple (example value = 100) to prefixes that should traverse the network over the delay-optimized path.

2. The route reflector receives the advertised route and advertises it to other PE nodes.

3. Ingress PEs in the network (such as node F) are pre-configured with an ODN template for color purple that provides the node with the steps to follow in case a route with the intended color appears, for example:
   - Contact SR-PCE and request computation for a path toward node H that does not share any nodes with another LSP in the same disjointness group.
   - At the head-end router, compute a path towards node H that minimizes cumulative delay.

4. In this example, the head-end router contacts the SR-PCE and requests computation for a path toward node H that minimizes cumulative delay.

5. After SR-PCE provides the compute path, an intent-driven SR policy is instantiated at the head-end router. Other prefixes with the same intent (color) and destined to the same egress PE can share the same on-demand SR policy. When the last prefix associated with a given [intent, egress PE] pair is withdrawn, the on-demand SR policy is deleted, and resources are freed from the head-end router.

An on-demand SR policy is created dynamically for BGP global or VPN (service) routes. The following services are supported with SR-ODN:
• IPv4 BGP global routes
• IPv6 BGP global routes (6PE)
• VPNv4
• VPNv6 (6vPE)
• EVPN-VPWS (single-homing)

**Configuration Steps**

To configure SR-ODN, complete the following configurations:

1. Define the SR-ODN template on the SR-TE head-end router.
   (Optional) If using Segment Routing Path Computation Element (SR-PCE) for path com- 
   putation:
   1. Configure SR-PCE. For detailed SR-PCE configuration information, see Configure SR-PCE, on page 130.
   2. Configure the head-end router as Path Computation Element Protocol (PCEP) Path Computation 
      Client (PCC). For detailed PCEP PCC configuration information, see Configure the Head-End Router 
      as PCEP PCC, on page 113.

2. Define BGP color extended communities. Refer to the "Implementing BGP" chapter in the Routing 

3. Define routing policies (using routing policy language [RPL]) to set BGP color extended 
   communities. Refer to the "Implementing Routing Policy" chapter in the Routing Configuration Guide for Cisco ASR 
   9000 Series Routers.

The following RPL attach-points for setting/matching BGP color extended communities are supported:

<table>
<thead>
<tr>
<th>Attach Point</th>
<th>Set</th>
<th>Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>VRF export</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VRF import</td>
<td>–</td>
<td>X</td>
</tr>
<tr>
<td>Neighbor-in</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Neighbor-out</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Inter-AFI export</td>
<td>–</td>
<td>X</td>
</tr>
<tr>
<td>Inter-AFI import</td>
<td>–</td>
<td>X</td>
</tr>
<tr>
<td>Default-originate</td>
<td>X</td>
<td>–</td>
</tr>
</tbody>
</table>
4. Apply routing policies to a service. Refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco ASR 9000 Series Routers*.

**Configure On-Demand Color Template**

- Use the `on-demand color color` command to create an ODN template for the specified color value. The head-end router automatically follows the actions defined in the template upon arrival of BGP global or VPN routes with a BGP color extended community that matches the color value specified in the template.

  The color range is from 1 to 4294967295.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color color
```

**Note**

Matching based on BGP color extended communities is performed automatically via ODN’s on-demand color template. RPL routing policies are not required.

- Use the `on-demand color color dynamic` command to associate the template with on-demand SR policies with a locally computed dynamic path (by SR-TE head-end router utilizing its TE topology database) or centrally (by SR-PCE). The head-end router will first attempt to install the locally computed path; otherwise, it will use the path computed by the SR-PCE.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color color dynamic
```

- Use the `on-demand color color dynamic pcep` command to indicate that only the path computed by SR-PCE should be associated with the on-demand SR policy. With this configuration, local path computation is not attempted; instead the head-end router will only instantiate the path computed by the SR-PCE.

```
Router(config-sr-te)# on-demand color color dynamic pcep
```

**Configure Dynamic Path Optimization Objectives**

- Use the `metric type {igp | te | latency}` command to configure the metric for use in path computation.

```
Router(config-sr-te-color-dyn)# metric type {igp | te | latency}
```

- Use the `metric margin {absolute value | relative percent}` command to configure the On-Demand dynamic path metric margin. The range for value and percent is from 0 to 2147483647.

```
Router(config-sr-te-color-dyn)# metric margin {absolute margin | relative margin}
```

**Configure Dynamic Path Constraints**

- Use the `disjoint-path group-id group-id type {link | node | srlg | srlg-node} [sub-id sub-id]` command to configure the disjoint-path constraints. The group-id and sub-id range is from 1 to 65535.

```
Router(config-sr-te-color-dyn)# disjoint-path group-id group-id type {link | node | srlg | srlg-node} [sub-id sub-id]
```
• Use the `affinity {include-any | include-all | exclude-any} {name WORD}` command to configure the affinity constraints.

```sh
Router(config-sr-te-color-dyn)＃ affinity {include-any | include-all | exclude-any} {name WORD}
```

• Use the `sid-algorithm algorithm-number` command to configure the SR Flexible Algorithm constraints. The `algorithm-number` range is from 128 to 255.

```sh
Router(config-sr-te-color-dyn)＃ sid-algorithm algorithm-number
```

• Use the `maximum-sid-depth value` command to customize the maximum SID depth (MSD) constraints advertised by the router.

  The default MSD value is equal to the maximum MSD supported by the platform (10).

```sh
Router(config-sr-te-color)＃ maximum-sid-depth value
```

A PCC can signal its MSD to the PCE in the following ways:

• During PCEP session establishment – The signaled MSD is treated as a node-wide property.
  - MSD is configured under `segment-routing traffic-eng maximum-sid-depth value` command

• During PCEP LSP path request – The signaled MSD is treated as an LSP property.
  - On-demand (ODN) SR Policy: MSD is configured using the `segment-routing traffic-eng on-demand color color maximum-sid-depth value` command
  - Local SR Policy: MSD is configured using the `segment-routing traffic-eng policy WORD candidate-paths preference preference dynamic metric sid-limit value` command.

After path computation, the PCE verifies whether the resulting label stack size complies with the MSD requirement. If the label stack size is larger than the MSD, then the PCE returns a "no path" response to the PCC.

### Configuring SR-ODN: Examples

The following examples show end-to-end configurations used in implementing SR-ODN on the head-end router.

#### Configuring ODN Color Templates: Example

Configure ODN color templates on routers acting as SR-TE head-end nodes. The following example shows various ODN color templates:

- color 10: minimization objective = te-metric
- color 20: minimization objective = igp-metric
- color 21: minimization objective = igp-metric; constraints = affinity
- color 30: minimization objective = delay-metric
- color 128: constraints = flex-algo
segment-routing
  traffic-eng
  on-demand color 10
   dynamic
    metric
      type te
    !
  !
  on-demand color 20
   dynamic
    metric
      type igp
    !
  !
  on-demand color 21
   dynamic
    metric
      type igp
      affinity exclude-any
       name CROSS
    !
  !
  on-demand color 30
   dynamic
    metric
      type latency
    !
  !
  on-demand color 128
   dynamic
    sid-algorithm 128
    !
  !
end

Configuring BGP Color Extended Community Set: Example

The following example shows how to configure BGP color extended communities that are later applied to BGP service routes via route-policies.

Note

In most common scenarios, egress PE routers that advertise BGP service routes apply (set) BGP color extended communities. However, color can also be set at the ingress PE router.

extcommunity-set opaque color10-te
  10
end-set
!
extcommunity-set opaque color20-igp
  20
end-set
!
extcommunity-set opaque color21-igp-excl-cross
  21
end-set
Configuring RPL to Set BGP Color (Layer-3 Services): Examples

The following example shows various representative RPL definitions that set BGP color community.

The first 4 RPL examples include the set color action only. The last RPL example performs the set color action for selected destinations based on a prefix-set.

```plaintext
route-policy SET_COLOR_LOW_LATENCY_TE
  set extcommunity color color10-te
  pass
end-policy

route-policy SET_COLOR_HI_BW
  set extcommunity color color20-igp
  pass
end-policy

route-policy SET_COLOR_LOW_LATENCY
  set extcommunity color color30-delay
  pass
end-policy

route-policy SET_COLOR_FA_128
  set extcommunity color color128-fa128
  pass
end-policy

prefix-set sample-set
  88.1.0.0/24
end-set

route-policy SET_COLOR_GLOBAL
  if destination in sample-set then
    set extcommunity color color10-te
  else
    pass
  endif
end-policy
```

Applying RPL to BGP Services (Layer-3 Services): Example

The following example shows various RPLs that set BGP color community being applied to BGP Layer-3 VPN services (VPNv4/VPNv6) and BGP global.

- The L3VPN examples show the RPL applied at the VRF export attach-point.
- The BGP global example shows the RPL applied at the BGP neighbor-out attach-point.

```plaintext
vrf vrf_cust1
  address-family ipv4 unicast
  export route-policy SET_COLOR_LOW_LATENCY_TE
```

address-family ipv6 unicast
  export route-policy SET_COLOR_LOW_LATENCY_TE

vrf vrf_cust2
  address-family ipv4 unicast
    export route-policy SET_COLOR_HI_BW
  address-family ipv6 unicast
    export route-policy SET_COLOR_HI_BW

vrf vrf_cust3
  address-family ipv4 unicast
    export route-policy SET_COLOR_LOW_LATENCY
  address-family ipv6 unicast
    export route-policy SET_COLOR_LOW_LATENCY

vrf vrf_cust4
  address-family ipv4 unicast
    export route-policy SET_COLOR_FA_128
  address-family ipv6 unicast
    export route-policy SET_COLOR_FA_128

router bgp 100
  neighbor-group BR-TO-RR
    address-family ipv4 unicast
      route-policy SET_COLOR_GLOBAL out

Verifying BGP VRF Information

Use the `show bgp vrf` command to display BGP prefix information for VRF instances. The following output shows the BGP VRF table including a prefix (88.1.1.0/24) with color 10 advertised by router 1.1.1.8.

RP/0/RP0/CPU0:R4# show bgp vrf vrf_cust1

BGP VRF vrf_cust1, state: Active
BGP Route Distinguisher: 1.1.1.4:101
VRF ID: 0xe00000007
BGP router identifier 1.1.1.4, local AS number 100
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe00000007  RD version: 282
BGP main routing table version 287
BGP NSR Initial initsync version 31 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Hop</th>
<th>Metric LocPrf</th>
<th>Weight Path</th>
</tr>
</thead>
</table>
*-> 44.1.1.0/24  | 40.4.101.11   | 0 400 (1) i   |             |
*-> 155.1.1.0/24 | 1.1.1.5       | 100           | 0 500 (1) i |
**-> 88.1.1.0/24 | 1.1.1.8 C:10  | 100           | 0 800 (1) i |
The following output displays the details for prefix 88.1.1.0/24. Note the presence of BGP extended color community 10, and that the prefix is associated with an SR policy with color 10 and BSID value of 24036.

RP/0/RP0/CPU0:R4# show bgp vrf vrf_cust1 88.1.1.0/24

BGP routing table entry for 88.1.1.0/24, Route Distinguisher: 1.1.1.4:101

Versions:
  Process: bRIB/RIB  SendTblVer
  Speaker: 282  282

Last Modified: May 20 09:23:34.112 for 00:06:03

Paths: (1 available, best #1)
  Advertised to CE peers (in unique update groups): 40.4.101.11
  Path #1: Received by speaker 0

Advertised to CE peers (in unique update groups):
  40.4.101.11
  800 (1)

1.1.1.8 C:10 (bsid:24036) (metric 20) from 1.1.1.55 (1.1.1.8)
  Received Label 24012
  Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported
  Received Path ID 0, Local Path ID 1, version 273
  Extended community: Color:10
  Originator: 1.1.1.8, Cluster list: 1.1.1.55
  SR policy color 10, up, registered, bsid 24036, if-handle 0x08000024

Source AFI: VPNv4 Unicast, Source VRF: default, Source Route Distinguisher: 1.1.1.8:101

---

Verifying Forwarding (CEF) Table

Use the show cef vrf command to display the contents of the CEF table for the VRF instance. Note that prefix 88.1.1.0/24 points to the BSID label corresponding to an SR policy. Other non-colored prefixes, such as 55.1.1.0/24, point to BGP next-hop.

RP/0/RP0/CPU0:R4# show cef vrf vrf_cust1

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0/0</td>
<td>drop</td>
<td>default handler</td>
</tr>
<tr>
<td>0.0.0.0/32</td>
<td>broadcast</td>
<td></td>
</tr>
<tr>
<td>40.4.101.0/24</td>
<td>attached</td>
<td>TenGigE0/0/0/0/101</td>
</tr>
<tr>
<td>40.4.101.0/32</td>
<td>broadcast</td>
<td>TenGigE0/0/0/0/101</td>
</tr>
<tr>
<td>40.4.101.4/32</td>
<td>receive</td>
<td>TenGigE0/0/0/0/101</td>
</tr>
<tr>
<td>40.4.101.11/32</td>
<td>40.4.101.11/32</td>
<td>TenGigE0/0/0/0/101</td>
</tr>
<tr>
<td>40.4.101.255/32</td>
<td>broadcast</td>
<td>TenGigE0/0/0/0/101</td>
</tr>
<tr>
<td>44.1.1.0/24</td>
<td>40.4.101.11/32</td>
<td>&lt;recursive&gt;</td>
</tr>
<tr>
<td>88.1.1.0/24</td>
<td>24036 (via-label)</td>
<td>&lt;recursive&gt;</td>
</tr>
<tr>
<td>55.1.1.0/24</td>
<td>1.1.1.5/32</td>
<td>&lt;recursive&gt;</td>
</tr>
<tr>
<td>99.1.1.0/24</td>
<td>1.1.1.9/32</td>
<td>&lt;recursive&gt;</td>
</tr>
<tr>
<td>224.0.0.0/4</td>
<td>0.0.0.0/32</td>
<td></td>
</tr>
<tr>
<td>224.0.0.0/24</td>
<td>receive</td>
<td></td>
</tr>
<tr>
<td>255.255.255.255/32</td>
<td>broadcast</td>
<td></td>
</tr>
</tbody>
</table>

The following output displays CEF details for prefix 88.1.1.0/24. Note that the prefix is associated with an SR policy with BSID value of 24036.
Verifying SR Policy

Use the `show segment-routing traffic-eng policy` command to display SR policy information.

The following outputs show the details of an on-demand SR policy that was triggered by prefixes with color 10 advertised by node 1.1.1.8.

```
RP/0/RP0/CPU0:R4# show segment-routing traffic-eng policy color 10 tabular

<table>
<thead>
<tr>
<th>Color</th>
<th>Endpoint</th>
<th>Admin State</th>
<th>Oper State</th>
<th>Binding SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.1.1.8</td>
<td>up</td>
<td>up</td>
<td>24036</td>
</tr>
</tbody>
</table>
```

The following outputs show the details of the on-demand SR policy for BSID 24036.

```
RP/0/RP0/CPU0:R4# show segment-routing traffic-eng policy binding-sid 24036

SR-TE policy database
---------------------
Color: 10, End-point: 1.1.1.8
Name: srte_c_10_ep_1.1.1.8
Status: Admin: up Operational: up for 4d14h (since Jul 3 20:28:57.840)
Candidate-paths:
Preference: 200 (BGP ODN) (active)
  Requested BSID: dynamic
  PCC info:
    Symbolic name: bgp_c_10_ep_1.1.1.8_discr_200
    PLSID-ID: 12
    Dynamic (valid)
    Metric Type: TE, Path Accumulated Metric: 30
    16009 [Prefix-SID, 1.1.1.9]
    16008 [Prefix-SID, 1.1.1.8]
Preference: 100 (BGP ODN)
  Requested BSID: dynamic
  PCC info:
    Symbolic name: bgp_c_10_ep_1.1.1.8_discr_100
    PLSID-ID: 11
    Dynamic (pce 1.1.1.57) (valid)
```

Note

There are 2 candidate paths associated with this SR policy: the path that is computed by the head-end router (with preference 200), and the path that is computed by the SR-PCE (with preference 100). The candidate path with the highest preference is the active candidate path (highlighted below) and is installed in forwarding.
Metric Type: TE, Path Accumulated Metric: 30
16009 [Prefix-SID, 1.1.1.9]
16008 [Prefix-SID, 1.1.1.8]

Attributes:
- **Binding SID**: 24036
- Forward Class: 0
- Steering BGP disabled: no
- IPv6 caps enable: yes

### Verifying SR Policy Forwarding

Use the `show segment-routing traffic-eng forwarding policy` command to display the SR policy forwarding information.

The following outputs show the forwarding details for an on-demand SR policy that was triggered by prefixes with color 10 advertised by node 1.1.1.8.

```
RP/0/RP0/CPU0:R4# show segment-routing traffic-eng forwarding policy binding-sid 24036 tabular

<table>
<thead>
<tr>
<th>Color</th>
<th>Endpoint</th>
<th>Segment List</th>
<th>Outgoing Label</th>
<th>Outgoing Interface</th>
<th>Next Hop</th>
<th>Bytes Switched</th>
<th>Pure Backup</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.1.1.8</td>
<td>dynamic</td>
<td>16009</td>
<td>Gi0/0/0/4</td>
<td>10.4.5.5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>16001</td>
<td>Gi0/0/0/5</td>
<td>11.4.8.8</td>
<td>0</td>
<td>Yes</td>
</tr>
</tbody>
</table>
```

```RP/0/RP0/CPU0:R4# show segment-routing traffic-eng forwarding policy binding-sid 24036 detail```

Mon Jul 8 11:56:46.887 PST

SR-TE Policy Forwarding database
---------------------------------
Color: 10, End-point: 1.1.1.8
Name: srte_c_10_ep_1.1.1.8
Binding SID: 24036
Segment Lists:
SL[0]:
- Name: dynamic
- Paths:
  - **Path[0]:**
    - Outgoing Label: 16009
    - Outgoing Interface: GigabitEthernet0/0/0/4
    - Next Hop: 10.4.5.5
    - Switched Packets/Bytes: 0/0
    - FRR Pure Backup: No
    - Label Stack (Top -> Bottom): (16009, 16008)
    - Path-id: 1 (Protected), Backup-path-id: 2, Weight: 64
  - **Path[1]:**
    - Outgoing Label: 16001
    - Outgoing Interface: GigabitEthernet0/0/0/5
    - Next Hop: 11.4.8.8
    - Switched Packets/Bytes: 0/0
    - FRR Pure Backup: Yes
    - Label Stack (Top -> Bottom): (16001, 16009, 16008)
    - Path-id: 2 (Pure-Backup), Weight: 64
Policy Packets/Bytes Switched: 0/0
Local label: 80013
```
Configuring SR-ODN for EVPN-VPWS: Use Case

This use case shows how to set up a pair of ELINE services using EVPN-VPWS between two sites. Services are carried over SR policies that must not share any common links along their paths (link-disjoint). The SR policies are triggered on-demand based on ODN principles. An SR-PCE computes the disjoint paths.

This use case uses the following topology with 2 sites: Site 1 with nodes A and B, and Site 2 with nodes C and D.

Figure 3: Topology for Use Case: SR-ODN for EVPN-VPWS

<table>
<thead>
<tr>
<th>IP Addresses of Loopback0 (Lo0) Interfaces</th>
<th>SR-PCE Lo0: 1.1.1.207</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site 1:</td>
<td></td>
</tr>
<tr>
<td>• Node A Lo0: 1.1.1.5</td>
<td></td>
</tr>
<tr>
<td>• Node B Lo0: 1.1.1.6</td>
<td></td>
</tr>
<tr>
<td>Site 2:</td>
<td></td>
</tr>
<tr>
<td>• Node C Lo0: 1.1.1.2</td>
<td></td>
</tr>
<tr>
<td>• Node D Lo0: 1.1.1.4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EVPN-VPWS Service Parameters</th>
<th>ELINE-1:</th>
<th>ELINE-2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site 1:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• EVPN-VPWS EVI 100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Node A: AC-ID = 11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Node C: AC-ID = 21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 2:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• EVPN-VPWS EVI 101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Node B: AC-ID = 12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Node D: AC-ID = 22</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ODN BGP Color Extended Communities</th>
<th>Site 1 routers (Nodes A and B):</th>
<th>Site 2 routers (Nodes C and D):</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• set color 10000</td>
<td>• set color 11000</td>
</tr>
<tr>
<td></td>
<td>• match color 11000</td>
<td>• match color 10000</td>
</tr>
</tbody>
</table>

Note: These colors are associated with the EVPN route-type 1 routes of the EVPN-VPWS services.
The use case provides configuration and verification outputs for all devices.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration: SR-PCE, on page 94</td>
<td>Verification: SR-PCE, on page 98</td>
</tr>
<tr>
<td>Configuration: Site 1 Node A, on page 94</td>
<td>Verification: Site 1 Node A, on page 102</td>
</tr>
<tr>
<td>Configuration: Site 1 Node B, on page 95</td>
<td>Verification: Site 1 Node B, on page 105</td>
</tr>
<tr>
<td>Configuration: Site 2 Node C, on page 96</td>
<td>Verification: Site 2 Node C, on page 108</td>
</tr>
<tr>
<td>Configuration: Site 2 Node D, on page 97</td>
<td>Verification: Site 2 Node D, on page 110</td>
</tr>
</tbody>
</table>

**Configuration: SR-PCE**

For cases when PCC nodes support, or signal, PCEP association-group object to indicate the pair of LSPs in a disjoint set, there is no extra configuration required at the SR-PCE to trigger disjoint-path computation.

---

**Note**

SR-PCE also supports disjoint-path computation for cases when PCC nodes do not support PCEP association-group object. See Configure the Disjoint Policy (Optional), on page 132 for more information.

**Configuration: Site 1 Node A**

This section depicts relevant configuration of Node A at Site 1. It includes service configuration, BGP color extended community, and RPL. It also includes the corresponding ODN template required to achieve the disjointness SLA.

Nodes in Site 1 are configured to set color 10000 on originating EVPN routes, while matching color 11000 on incoming EVPN routes from routers located at Site 2.

Since both nodes in Site 1 request path computation from SR-PCE using the same disjoint-path group-id (775), the PCE will attempt to compute disjointness for the pair of LSPs originating from Site 1 toward Site 2.

```c
/* EVPN-VPWS configuration */
interface GigabitEthernet0/0/0/3.2500 l2transport
    encapsulation dot1q 2500
    rewrite ingress tag pop 1 symmetric
l2vpn
    xconnect group evpn_vpws_group
    p2p evpn_vpws_100
    interface GigabitEthernet0/0/0/3.2500
    neighbor evpn evi 100 target 21 source 11
```
/* BGP color community and RPL configuration */

```conf
extcommunity-set opaque color=10000
10000
end-set
```

```conf
route-policy SET_COLOR_EVPN_VPWS
  if evpn-route-type is 1 and rd in (ios-regex '.*..*..*..*:100') then
    set extcommunity color=color-10000
  endif
pass
end-policy
! router bgp 65000
  neighbor 1.1.1.253
    address-family l2vpn evpn
    route-policy SET_COLOR_EVPN_VPWS out
  !
! /* ODN template configuration */

```conf
segment-routing
  traffic-eng
    on-demand color=11000
    dynamic
      pcep
      ! metric
type igp
    ! disjoint-path group-id 775 type link
    !
!
```

**Configuration: Site 1 Node B**

This section depicts relevant configuration of Node B at Site 1.

```conf
/* EVPN-VPWS configuration */

interface TenGigE0/3/0/0/8.2500 l2transport
  encapsulation dot1q 2500
  rewrite ingress tag pop 1 symmetric
! l2vpn
  xconnect group evpn_vpws_group
    p2p evpn_vpws_101
      interface TenGigE0/3/0/0/8.2500
        neighbor evpn evi 101 target 22 source 12
        !
    !
! /* BGP color community and RPL configuration */

extcommunity-set opaque color=10000
10000
```
end-set  !

route-policy SET_COLOR_EVPN_VPWS
  if evpn-route-type is 1 and rd in (ios-regex '^..*..*..*:101$') then
    set extcommunity color color-10000
  endif
pass
end-policy
!
router bgp 65000
  neighbor 1.1.1.253
    address-family l2vpn evpn
      route-policy SET_COLOR_EVPN_VPWS out
    !
  !
/* ODN template configuration */

segment-routing
  traffic-eng
    on-demand color 11000
      dynamic
      pcep
      !
      metric
      type igp
      !
      disjoint-path group-id 775 type link
    !
    !
  !

**Configuration: Site 2 Node C**

This section depicts relevant configuration of Node C at Site 2. It includes service configuration, BGP color extended community, and RPL. It also includes the corresponding ODN template required to achieve the disjointness SLA.

Nodes in Site 2 are configured to set color 11000 on originating EVPN routes, while matching color 10000 on incoming EVPN routes from routers located at Site 1.

Since both nodes on Site 2 request path computation from SR-PCE using the same disjoint-path group-id (776), the PCE will attempt to compute disjointness for the pair of LSPs originating from Site 2 toward Site 1.

/* EVPN-VPWS configuration */

interface GigabitEthernet0/0/0/3.2500 l2transport
  encapsulation dot1q 2500
  rewrite ingress tag pop 1 symmetric
  !
  l2vpn
  xconnect group evpn_vpws_group
  p2p evpn_vpws_100
  interface GigabitEthernet0/0/0/3.2500
    neighbor evpn evi 100 target 11 source 21
    !
    !
    !
/* BGP color community and RPL configuration */

extcommunity-set opaque color-11000

11000
end-set

route-policy SET_COLOR_EVPN_VPWS
if evpn-route-type is 1 and rd in (ios-regex '.*.*.*.*:(100)') then
    set extcommunity color color-11000
endif
pass
end-policy

router bgp 65000
neighbor 1.1.1.253
    address-family l2vpn evpn
        route-policy SET_COLOR_EVPN_VPWS out

/* ODN template configuration */

segment-routing
traffic-eng
    on-demand color 10000
        dynamic
            pcep
            metric
                type igp
                    disjoint-path group-id 776 type link
            !

Configuration: Site 2 Node D

This section depicts relevant configuration of Node D at Site 2.

/* EVPN-VPWS configuration */

interface GigabitEthernet0/0/0/1.2500 l2transport
    encapsulation dot1q 2500
    rewrite ingress tag pop 1 symmetric

l2vpn
    xconnect group evpn_vpws_group
        p2p evpn_vpws_101
            interface GigabitEthernet0/0/0/1.2500
                neighbor evpn evi 101 target 12 source 22
        !

/* BGP color community and RPL configuration */

extcommunity-set opaque color-11000

11000
end-set

!
route-policy SET_COLOR_EVPN_VPWS
if evpn-route-type is 1 and rd in (ios-regex '.*::*::*::*:((101))') then
    set extcommunity color color-11000
endif
pass
end-policy

router bgp 65000
neighbor 1.1.1.253
    address-family l2vpn evpn
    route-policy SET_COLOR_EVPN_VPWS out

/* ODN template configuration */
segment-routing
    traffic-eng
        on-demand color 10000
        dynamic
            pcep
            metric
                type igp
                disjoint-path group-id 776 type link

Verification: SR-PCE

Use the show pce ipv4 peer command to display the SR-PCE's PCEP peers and session status. SR-PCE performs path computation for the 4 nodes depicted in the use-case.

RP/0/0/CP00:SR-PCE# show pce ipv4 peer
Mon Jul 15 19:41:43.622 UTC
PCE's peer database:
---------------------
Peer address: 1.1.1.2
    State: Up
    Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 1.1.1.4
    State: Up
    Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 1.1.1.5
    State: Up
    Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 1.1.1.6
    State: Up
    Capabilities: Stateful, Segment-Routing, Update, Instantiation

Use the show pce association group-id command to display information for the pair of LSPs assigned to a given association group-id value.
Based on the goals of this use case, SR-PCE computes link-disjoint paths for the SR policies associated with a pair of ELINE services between site 1 and site 2. In particular, disjoint LSPs from site 1 to site 2 are identified by association group-id 775. The output includes high-level information for LSPs associated to this group-id:

- At Node A (1.1.1.5): LSP symbolic name = bgp_c_11000_ep_1.1.1.2_discr_100
- At Node B (1.1.1.6): LSP symbolic name = bgp_c_11000_ep_1.1.1.4_discr_100

In this case, the SR-PCE was able to achieve the desired disjointness level; therefore the Status is shown as "Satisfied".

```
RP/0/0/CPU0:SR-PCE# show pce association group-id 775
Thu Jul 11 03:52:20.770 UTC
PCE's association database:
----------------------
Association: Type Link-Disjoint, Group 775, Not Strict
Associated LSPs:
LSP[0]:
  PCC 1.1.1.6, tunnel name bgp_c_11000_ep_1.1.1.4_discr_100, PLSP ID 18, tunnel ID 17, LSP ID 3, Configured on PCC
LSP[1]:
  PCC 1.1.1.5, tunnel name bgp_c_11000_ep_1.1.1.2_discr_100, PLSP ID 18, tunnel ID 18, LSP ID 3, Configured on PCC
Status: Satisfied
```

Use the `show pce lsp` command to display detailed information of an LSP present in the PCE's LSP database. This output shows details for the LSP at Node A (1.1.1.5) that is used to carry traffic of EVPN VPWS EVI 100 towards node C (1.1.1.2).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 1.1.1.5 name bgp_c_11000_ep_1.1.1.2_discr_100
Thu Jul 11 03:58:45.903 UTC
PCE's tunnel database:
----------------------
PCC 1.1.1.5:
Tunnel Name: bgp_c_11000_ep_1.1.1.2_discr_100
Color: 11000
Interface Name: srte_c_11000_ep_1.1.1.2
LSPs:
LSP[0]:
  source 1.1.1.5, destination 1.1.1.2, tunnel ID 18, LSP ID 3
  State: Admin up, Operation up
  Setup type: Segment Routing
  Binding SID: 80037
  Maximum SID Depth: 10
  Absolute Metric Margin: 0
  Relative Metric Margin: 0%
  Preference: 100
  Bandwidth: signaled 0 kbps, applied 0 kbps
  PCEP information:
    PLSP-ID 0x12, flags: D:1 S:0 R:0 A:1 O:1 C:0
    LSP Role: Exclude LSP
    State-sync PCE: None
    PCC: 1.1.1.5
    LSP is subdelegated to: None
  Reported path:
    Metric type: IGP, Accumulated Metric 40
    SID[0]: Adj, Label 80003, Address: local 11.5.8.5 remote 11.5.8.8
    SID[1]: Node, Label 16007, Address 1.1.1.7
    SID[2]: Node, Label 16002, Address 1.1.1.2
    Computed path: (Local PCE)
```
This output shows details for the LSP at Node B (1.1.1.6) that is used to carry traffic of EVPN VPWS EVI 101 towards node D (1.1.1.4).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 1.1.1.6 name bgp_c_11000_ep_1.1.1.4_discr_100
Thu Jul 11 03:58:56.812 UTC
```

PCE's tunnel database:
----------------------
PCC 1.1.1.6:
Tunnel Name: bgp_c_11000_ep_1.1.1.4_discr_100
Color: 11000
Interface Name: srte_c_11000_ep_1.1.1.4
LSPs:
LSP[0]:
  source 1.1.1.6, destination 1.1.1.4, tunnel ID 17, LSP ID 3
  State: Admin up, Operation up
  Setup type: Segment Routing
  Binding SID: 80061
  Maximum SID Depth: 10
  Absolute Metric Margin: 0
  Relative Metric Margin: 0%
  Preference: 100
  Bandwidth: signaled 0 kbps, applied 0 kbps
PCEP information:
  PLSP-ID 0x12, flags: D:1 S:0 R:0 A:1 O:1 C:0
  LSP Role: Disjoint LSP
  State-sync PCE: None
  PCC: 1.1.1.6
  LSP is subdelegated to: None
Reported path:
  Metric type: IGP, Accumulated Metric 40
  SID[0]: Adj, Label 80003, Address: local 11.5.8.5 remote 11.5.8.8
  SID[1]: Node, Label 16007, Address 1.1.1.7
  SID[2]: Node, Label 16002, Address 1.1.1.2
```

Disjoint Group Information:
Type Link-Disjoint, Group 775

Based on the goals of this use case, SR-PCE computes link-disjoint paths for the SR policies associated with a pair of ELINE services between site 1 and site 2. In particular, disjoint LSPs from site 2 to site 1 are identified by association group-id 776. The output includes high-level information for LSPs associated to this group-id:

- At Node C (1.1.1.2): LSP symbolic name = bgp_c_10000_ep_1.1.1.5_discr_100
- At Node D (1.1.1.4): LSP symbolic name = bgp_c_10000_ep_1.1.1.6_discr_100
In this case, the SR-PCE was able to achieve the desired disjointness level; therefore, the Status is shown as "Satisfied".

RP/0/0/CPU0:SR-PCE# show pce association group-id 776
Thu Jul 11 03:52:24.370 UTC

PCE's association database:
----------------------
**Association: Type Link-Disjoint, Group 776, Not Strict**
Associated LSPs:
LSP[0]:
  PCC 1.1.1.4, tunnel name bgp_c_10000_ep_1.1.1.6_discr_100, PLSP ID 16, tunnel ID 14, LSP ID 1, Configured on PCC
LSP[1]:
  PCC 1.1.1.2, tunnel name bgp_c_10000_ep_1.1.1.5_discr_100, PLSP ID 6, tunnel ID 21, LSP ID 3, Configured on PCC

**Status: Satisfied**

Use the show pce lsp command to display detailed information of an LSP present in the PCE's LSP database. This output shows details for the LSP at Node C (1.1.1.2) that is used to carry traffic of EVPN VPWS EVI 100 towards node A (1.1.1.5).

RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 1.1.1.2 name bgp_c_10000_ep_1.1.1.5_discr_100
Thu Jul 11 03:55:21.706 UTC

PCE's tunnel database:
----------------------
PCC 1.1.1.2:
Tunnel Name: bgp_c_10000_ep_1.1.1.5_discr_100
Color: 10000
**Interface Name: srte_c_10000_ep_1.1.1.5**
LSPs:
LSP[0]:
  source 1.1.1.2, destination 1.1.1.5, tunnel ID 21, LSP ID 3
  State: Admin up, Operation up
  Setup type: Segment Routing
  Binding SID: 80052
  Maximum SID Depth: 10
  Absolute Metric Margin: 0
  Relative Metric Margin: 0%
  Preference: 100
  Bandwidth: signaled 0 kbps, applied 0 kbps
  PCEP information:
    PLSP-ID 0x6, flags: D:1 S:0 R:0 A:1 O:1 C:0
    LSP Role: Exclude LSP
    State-sync PCE: None
    PCC: 1.1.1.2
    LSP is subdelegated to: None
Reported path:
  Metric type: IGP, Accumulated Metric 40
  SID[0]: Node, Label 16007, Address 1.1.1.7
  SID[1]: Node, Label 16008, Address 1.1.1.8
  SID[2]: Adj, Label 80005, Address: local 11.5.8.8 remote 11.5.8.5
Computed path: (Local PCE)
  Computed Time: Thu Jul 11 03:50:03 UTC 2019 (00:05:18 ago)
  Metric type: IGP, Accumulated Metric 40
  SID[0]: Node, Label 16007, Address 1.1.1.7
  SID[1]: Node, Label 16008, Address 1.1.1.8
  SID[2]: Adj, Label 80005, Address: local 11.5.8.8 remote 11.5.8.5
Recorded path:
  None

**Disjoint Group Information:**
  **Type Link-Disjoint, Group 776**
This output shows details for the LSP at Node D (1.1.1.4) used to carry traffic of EVPN VPWS EVI 101 towards node B (1.1.1.6).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 1.1.1.4 name bgp_c_10000_ep_1.1.1.6_discr_100
Thu Jul 11 03:55:23.296 UTC

PCE's tunnel database:
-------------------------------------
PCC 1.1.1.4:
Tunnel Name: bgp_c_10000_ep_1.1.1.6_discr_100
Color: 10000
Interface Name: srte_c_10000_ep_1.1.1.6
LSPs:
LSP[0]:
  source 1.1.1.4, destination 1.1.1.6, tunnel ID 14, LSP ID 1
  State: Admin up, Operation up
  Setup type: Segment Routing
  Binding SID: 80047
  Maximum SID Depth: 10
  Absolute Metric Margin: 0
  Relative Metric Margin: 0%
  Preference: 100
  Bandwidth: signaled 0 kbps, applied 0 kbps
  PCEP information:
    PLSP-ID 0x10, flags: D:1 S:0 R:0 A:1 O:1 C:0
    LSP Role: Disjoint LSP
    State-sync PCE: None
    PCC: 1.1.1.4
    LSP is subdelegated to: None
  Reported path:
    Metric type: IGP, Accumulated Metric 40
    SID[0]: Node, Label 16001, Address 1.1.1.1
    SID[1]: Node, Label 16006, Address 1.1.1.6
  Computed path: (Local PCE)
    Computed Time: Thu Jul 11 03:50:03 UTC 2019 (00:05:20 ago)
    Metric type: IGP, Accumulated Metric 40
    SID[0]: Node, Label 16001, Address 1.1.1.1
    SID[1]: Node, Label 16006, Address 1.1.1.6
  Recorded path:
    None

Disjoint Group Information:
  Type Link-Disjoint, Group 776
```

**Verification: Site 1 Node A**

This section depicts verification steps at Node A.

Use the `show bgp l2vpn evpn` command to display BGP prefix information for EVPN-VPWS EVI 100 (rd 1.1.1.5:100). The output includes an EVPN route-type 1 route with color 11000 originated at Node C (1.1.1.2).

```
RP/0/RSP0/CPU0:Node-A# show bgp l2vpn evpn rd 1.1.1.5:100
Wed Jul 10 18:57:57.704 PST
BGP router identifier 1.1.1.5, local AS number 65000
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0  RD version: 0
BGP main routing table version 360
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs
Status codes: s suppressed, d damped, h history, * valid, > best
```
The following output displays the details for the incoming EVPN RT1. Note the presence of BGP extended color community 11000, and that the prefix is associated with an SR policy with color 11000 and BSID value of 80044.

RP/0/RSP0/CPU0:Node-A# show bgp l2vpn evpn rd 1.1.1.5:100
[1][0000.0000.0000.0000.0000][21]/120
Wed Jul 10 18:57:58.107 PST
BGP routing table entry for [1][0000.0000.0000.0000.0000][21]/120, Route Distinguisher: 1.1.1.5:100
Versions:
\[ process bRIB/RIB SendTblVer \]
Speaker 360 360
Last Modified: Jul 10 18:36:18.369 for 00:21:40
Paths: (1 available, best #1)
\[ Not advertised to any peer \]
Local
\[ 1.1.1.2 C:11000 (bsid:80044) (metric 40) from 1.1.1.253 (1.1.1.2) \]
Received Label 80056
Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported, rib-install
Received Path ID 0, Local Path ID 1, version 358
Extended community: Color:11000
Originator: 1.1.1.2, Cluster list: 1.1.1.253
SR policy color 11000, up, registered, bsid 80044, if-handle 0x00001b20
Source AFI: L2VPN EVVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.2:100

Use the `show l2vpn xconnect` command to display the state associated with EVPN-VPWS EVI 100 service.

RP/0/RSP0/CPU0:Node-A# show l2vpn xconnect group evpn_vpws_group
detail
Wed Jul 10 18:58:02.755 PST
Group evpn_vpws_group, XC evpn_vpws_100, state is up; Interworking none
AC: GigabitEthernet0/0/0/3.2500

The following output shows the details for the service. Note that the service is associated with the on-demand SR policy with color 11000 and end-point 1.1.1.2 (node C).

RP/0/RSP0/CPU0:Node-A# show l2vpn xconnect group evpn_vpws_group xc-name evpn_vpws_100
detail
Wed Jul 10 18:58:02.755 PST
Group evpn_vpws_group, XC evpn_vpws_100, state is up; Interworking none
AC: GigabitEthernet0/0/0/3.2500, state is up
Type VLAN; Num Ranges: 1
Rewrite Tags: []
VLAN ranges: [2500, 2500]
MTU 1500; XC ID 0x120000c; interworking none
Statistics:
   packets: received 0, sent 0
   bytes: received 0, sent 0
   drops: illegal VLAN 0, illegal length 0
EVPN: neighbor 1.1.1.2, FW ID: evi 100, ac-id 21, state is up (established)
   XC ID 0xa0000007
Encapsulation MPLS
Source address 1.1.1.5
Encap type Ethernet, control word enabled
Sequencing not set
Preferred path Active: SR TE srte_c_11000_ep_1.1.1.2, On-Demand, fallback enabled
Tunnel: Up
Load Balance Hashing: src-dst-mac

<table>
<thead>
<tr>
<th>EVPN</th>
<th>Local</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>80040</td>
<td>80056</td>
</tr>
<tr>
<td>MTU</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Control word enabled</td>
<td>enabled</td>
<td>enabled</td>
</tr>
<tr>
<td>AC ID</td>
<td>11</td>
<td>21</td>
</tr>
<tr>
<td>EVPN type</td>
<td>Ethernet</td>
<td>Ethernet</td>
</tr>
</tbody>
</table>

Create time: 10/07/2019 18:31:30 (1d17h ago)
Last time status changed: 10/07/2019 19:42:00 (1d16h ago)
Last time PW went down: 10/07/2019 19:40:55 (1d16h ago)
Statistics:
   packets: received 0, sent 0
   bytes: received 0, sent 0

Use the `show segment-routing traffic-eng policy` command with `tabular` option to display SR policy summary information.

The following output shows the on-demand SR policy with BSID 80044 that was triggered by EVPN RT1 prefix with color 11000 advertised by node C (1.1.1.2).

```
RP/0/RSP0/CPU0:Node-A# show segment-routing traffic-eng policy color 11000 tabular
Wed Jul 10 18:58:00.732 PST

Color Endpoint Admin Oper Binding SID
------ ---------- ----- ------ ------
11000 1.1.1.2 up up 80044
```

The following output shows the details for the on-demand SR policy. Note that the SR policy's active candidate path (preference 100) is computed by SR-PCE (1.1.1.207).

Based on the goals of this use case, SR-PCE computes link-disjoint paths for the SR policies associated with a pair of ELINE services between site 1 and site 2. Specifically, from site 1 to site 2, LSP at Node A (srte_c_11000_ep_1.1.1.2) is link-disjoint from LSP at Node B (srte_c_11000_ep_1.1.1.4).

```
RP/0/RSP0/CPU0:Node-A# show segment-routing traffic-eng policy color 11000
Wed Jul 10 19:15:47.217 PST

SR-TE policy database

Color: 11000, End-point: 1.1.1.2
```
**Name:** srte_c_11000_ep_1.1.1.2
**Status:**
Admin: up  Operational: up for 00:39:31 (since Jul 10 18:36:00.471)

**Candidate-paths:**
Preference: 200 (BGP ODN) (shutdown)
Requested BSID: dynamic
PCC info:
Symbolic name: bgp_c_11000_ep_1.1.1.2_discr_200
PLSP-ID: 19
Dynamic (invalid)

**Preference: 100 (BGP ODN) (active)**
Requested BSID: dynamic
PCC info:
Symbolic name: bgp_c_11000_ep_1.1.1.2_discr_100
PLSP-ID: 19
Dynamic (pce 1.1.1.207) (valid)

**Metric Type:** IGP  Path Accumulated Metric: 40
80003 [Adjacency-SID, 11.5.8.5 - 11.5.8.8]
16007 [Prefix-SID, 1.1.1.7]
16002 [Prefix-SID, 1.1.1.2]

**Attributes:**
- Binding SID: 80044
- Forward Class: 0
- Steering BGP disabled: no
- IPv6 caps enable: yes

**Verification: Site 1 Node B**

This section depicts verification steps at Node B.

Use the `show bgp l2vpn evpn` command to display BGP prefix information for EVPN-VPWS EVI 101 (rd 1.1.1.6:101). The output includes an EVPN route-type 1 route with color 11000 originated at Node D (1.1.1.4).

```
RP/0/RSP0/CPU0:Node-B# show bgp l2vpn evpn rd 1.1.1.6:101
Wed Jul 10 19:08:54.964 PST
BGP router identifier 1.1.1.6, local AS number 65000
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0  RD version: 0
BGP main routing table version 322
BGP NSR Initial initsync version 7 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
   i - internal, r RIB-failure, S stale, N Nexthop-discord
Origin codes: i - IGP, e - EGP, ? - incomplete
Network  Next Hop  Metric LocPrf Weight Path
Route Distinguisher: 1.1.1.6:101 (default for vrf VPWS:101)
   * [1][0000.0000.0000.0000.0000.0000.0000.0000] [12] /120
   0.0.0.0  0 i
   *>[1][0000.0000.0000.0000.0000.0000.0000.0000] [22] /120
   1.1.1.4 C:11000  100  0 i

Processed 2 prefixes, 2 paths
```

The following output displays the details for the incoming EVPN RT1. Note the presence of BGP extended color community 11000, and that the prefix is associated with an SR policy with color 11000 and BSID value of 80061.
Use the `show l2vpn xconnect` command to display the state associated with EVPN-VPWS EVI 101 service.

The following output shows the details for the service. Note that the service is associated with the on-demand SR policy with color 11000 and end-point 1.1.1.4 (node D).

**Show BGP Table Entry for RD 1.1.1.6:101**

```
WED JUL 10 19:08:56.388 PST
Legend: ST = State, UD = Up, DN = Down, AD = Admin Down, UR = Unresolved, SB = Standby, SR = Standby Ready, (PF) = Partially Programmed
XConnect Group Name ST Description ST Description ST
-------------------------------------------------------------
evpn_vpws_group evpn_vpws_101 UP Te0/3/0/0/8.2500 UP EVPN 101,2,1.1.1.4 UP
-------------------------------------------------------------
```

The following output shows the details for the service. Note that the service is associated with the on-demand SR policy with color 11000 and end-point 1.1.1.4 (node D).

```
RP/0/RSP0/CPU0:Node-B# show l2vpn xconnect group evpn_vpws_group xc-name evpn_vpws_101
Wed Jul 10 19:08:56.511 PST
Group evpn_vpws_group, XC evpn_vpws_101, state is up; Interworking none
AC: TenGigE0/3/0/0/8.2500, state is up
  Type VLAN; Num Ranges: 1
  Rewrite Tags: []
  VLAN ranges: [2500, 2500]
  MTU 1500; XC ID 0x2a0000e; interworking none
  Statistics:
    packets: received 0, sent 0
    bytes: received 0, sent 0
    drops: illegal VLAN 0, illegal length 0
  EVPN: neighbor 1.1.1.4, FW ID: evi 101, ac-id 22, state is up ( established )
  XC ID 0xa0000009
  Encapsulation MPLS
  Source address 1.1.1.6
  Encap type Ethernet, control word enabled
  Sequencing not set
  Preferred path Active : SR TE srte_c_11000_ep_1.1.1.4, On-Demand, fallback enabled
```
Use the `show segment-routing traffic-eng policy` command with the `tabular` option to display SR policy summary information.

The following output shows the on-demand SR policy with BSID 80061 that was triggered by EVPN RT1 prefix with color 11000 advertised by node D (1.1.1.4).

```
RP/0/RSP0/CPU0:Node-B# show segment-routing traffic-eng policy color 11000 tabular
```

```
Color Endpoint Admin Oper Binding
------- ------------------ ------ ------ -------------------
11000 1.1.1.4  up  up  80061
```

The following output shows the details for the on-demand SR policy. Note that the SR policy's active candidate path (preference 100) is computed by SR-PCE (1.1.1.207).

```
RP/0/RSP0/CPU0:Node-B# show segment-routing traffic-eng policy color 11000
```

```
SR-TE policy database
---------------------
Color: 11000, End-point: 1.1.1.4
Name: srte_c_11000_ep_1.1.1.4
Status:
 Admin: up Operational: up for 00:26:47 (since Jul 10 18:40:05.868)
Candidate-paths:
 Preference: 200 (BGP ODN) (shutdown)
 Requested BSID: dynamic
 PCC info:
 Symbolic name: bgp_c_11000_ep_1.1.1.4_discr_200
 PLSP-ID: 19
 Dynamic (invalid)
 Preference: 100 (BGP ODN) (active)
 Requested BSID: dynamic
 PCC info:
 Symbolic name: bgp_c_11000_ep_1.1.1.4_discr_100
 PLSP-ID: 18
 Dynamic (pce 1.1.1.207) (valid)
```
Metric Type: IGP, Path Accumulated Metric: 40
16001 [Prefix-SID, 1.1.1.1]
16004 [Prefix-SID, 1.1.1.4]

Attributes:
- **Binding SID**: 80061
- Forward Class: 0
- Steering BGP disabled: no
- IPv6 caps enable: yes

**Verification: Site 2 Node C**

This section depicts verification steps at Node C.

Use the `show bgp l2vpn evpn` command to display BGP prefix information for EVPN-VPWS EVI 100 (rd 1.1.1.2:100). The output includes an EVPN route-type 1 route with color 10000 originated at Node A (1.1.1.5).

```
RP/0/RSP0/CPU0:Node-C# show bgp l2vpn evpn rd 1.1.1.2:100
BGP router identifier 1.1.1.2, local AS number 65000
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 0
BGP main routing table version 21
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
Origin codes: i - IGP, e - EGP, ? - incomplete
Network Next Hop Metric LocPrf Weight Path
Route Distinguisher: 1.1.1.2:100 (default for vrf VPWS:100)
*>i[1][0000.0000.0000.0000.0000][11]/120
    1.1.1.5 C:10000 100 0 i
*> [1][0000.0000.0000.0000.0000][21]/120
    0.0.0.0 0 i
```

The following output displays the details for the incoming EVPN RT1. Note the presence of BGP extended color community 10000, and that the prefix is associated with an SR policy with color 10000 and BSID value of 80058.

```
RP/0/RSP0/CPU0:Node-C# show bgp l2vpn evpn rd 1.1.1.2:100
[1][0000.0000.0000.0000.0000][11]/120
BGP routing table entry for [1][0000.0000.0000.0000.0000][11]/120, Route Distinguisher:
1.1.1.2:100
Versions:
- Process bRIB/RIB SendTblVer
- Speaker 20 20
Last Modified: Jul 10 18:36:20.503 for 00:45:21
Paths: (1 available, best #1)
- Not advertised to any peer
- Path #1: Received by speaker 0
- Not advertised to any peer
- Local
  1.1.1.5 C:10000 (bsid:80058) (metric 40) from 1.1.1.253 (1.1.1.5)
  Received Label 80040
  Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported, rib-install
  Received Path ID 0, Local Path ID 1, version 18
  Extended community: Color:10000 RT:65000:100
  Originator: 1.1.1.5, Cluster list: 1.1.1.253
  SR policy color 10000, up, registered, bsid 80058, if-handle 0x000006a0
```
Use the `show l2vpn xconnect` command to display the state associated with EVPN-VPWS EVI 100 service.

```
RP/0/RSP0/CP00:Node-C# show l2vpn xconnect group evpn_vpws_group
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed
XConnect Group Name ST Description ST Description ST
------------------------ ----------------------------- -----------------------------
evpn_vpws_group evpn_vpws_100 UP EVPN 100,11,1.1.1.5 UP
----------------------------------------------------------------------------------------
The following output shows the details for the service. Note that the service is associated with the on-demand SR policy with color 10000 and end-point 1.1.1.5 (node A).

```
RP/0/RSP0/CP00:Node-C# show l2vpn xconnect group evpn_vpws_group xc-name evpn_vpws_100

Group evpn_vpws_group, XC evpn_vpws_100, state is up; Interworking none
AC: Gigabitethernet0/0/0/3.2500, state is up
  Type VLAN; Num Ranges: 1
  Rewrite Tags: []
  VLAN ranges: [2500, 2500]
  MTU 1500; XC ID 0x12000008; interworking none
Statistics:
  packets: received 0, sent 0
  bytes: received 0, sent 0
  drops: illegal VLAN 0, illegal length 0
EVPN: neighbor 1.1.1.5, PW ID: evi 100, ac-id 11, state is up ( established )
  XC ID 0xa0000003
  Encapsulation MPLS
  Source address 1.1.1.2
  Encap type Ethernet, control word enabled
  Sequencing not set
Preferred path Active : SR TE srte_c_10000_ep_1.1.1.5, On-Demand, fallback enabled
Tunnel : Up
Load Balance Hashing: src-dst-mac

<table>
<thead>
<tr>
<th>EVPN</th>
<th>Local</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>80056</td>
<td>80040</td>
</tr>
<tr>
<td>MTU</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Control word enabled</td>
<td>enabled</td>
<td>enabled</td>
</tr>
<tr>
<td>AC ID</td>
<td>21</td>
<td>11</td>
</tr>
<tr>
<td>EVPN type</td>
<td>Ethernet</td>
<td>Ethernet</td>
</tr>
</tbody>
</table>

Create time: 10/07/2019 18:36:16 (1d19h ago)
Last time status changed: 10/07/2019 19:41:59 (1d18h ago)
Last time PW went down: 10/07/2019 19:40:54 (1d18h ago)
Statistics:
  packets: received 0, sent 0
  bytes: received 0, sent 0
```

Use the `show segment-routing traffic-eng policy` command with `tabular` option to display SR policy summary information.
The following output shows the on-demand SR policy with BSID 80058 that was triggered by EVPN RT1 prefix with color 10000 advertised by node A (1.1.1.5).

```
RP/0/RSP0/CPU0:Node-C# show segment-routing traffic-eng policy color 10000 tabular
```

<table>
<thead>
<tr>
<th>Color</th>
<th>Endpoint</th>
<th>Admin</th>
<th>Oper</th>
<th>Binding SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>1.1.1.5</td>
<td>up</td>
<td>up</td>
<td>80058</td>
</tr>
</tbody>
</table>

The following output shows the details for the on-demand SR policy. Note that the SR policy’s active candidate path (preference 100) is computed by SR-PCE (1.1.1.207).

Based on the goals of this use case, SR-PCE computes link-disjoint paths for the SR policies associated with a pair of ELINE services between site 1 and site 2. Specifically, from site 2 to site 1, LSP at Node C (srte_c_10000_ep_1.1.1.5) is link-disjoint from LSP at Node D (srte_c_10000_ep_1.1.1.6).

```
RP/0/RSP0/CPU0:Node-C# show segment-routing traffic-eng policy color 10000
```

SR-TE policy database

```
Color: 10000, End-point: 1.1.1.5
Name: srte_c_10000_ep_1.1.1.5
Status:
  Admin: up Operational: up for 00:12:35 (since Jul 10 19:49:21.890)
Candidate-paths:
  Preference: 100 (BGP ODN) (active)
    Requested BSID: dynamic
    PCC info:
      Symbolic name: bgp_c_10000_ep_1.1.1.5_discr_100
      PLSID: 6
      Dynamic (valid)

  Preference: 200 (BGP ODN) (shutdown)
    Requested BSID: dynamic
    PCC info:
      Symbolic name: bgp_c_10000_ep_1.1.1.5_discr_200
      PLSID: 7
      Dynamic (invalid)
```

```
Verification: Site 2 Node D
```

This section depicts verification steps at Node D.

Use the `show bgp l2vpn evpn` command to display BGP prefix information for EVPN-VPWS EVI 101 (rd 1.1.1.4:101). The output includes an EVPN route-type 1 route with color 10000 originated at Node B (1.1.1.6).

```
RP/0/RSP0/CPU0:Node-D# show bgp l2vpn evpn rd 1.1.1.4:101
```
BGP main routing table version 570
BGP NSR initial intsycr version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
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
Network   Next Hop       Metric LocPrf Weight Path
Route Distinguisher: 1.1.1.4:101 (default for vrf VPWS:101)
*>i[1][0000.0000.0000.0000.0000][12]/120
\n1.1.1.6 C:10000
100  0  i
*> [1][0000.0000.0000.0000.0000][22]/120
  0.0.0.0  0  i

Processed 2 prefixes, 2 paths

The following output displays the details for the incoming EVPN RT1. Note the presence of BGP extended color community 10000, and that the prefix is associated with an SR policy with color 10000 and BSID value of 80047.

RP/0/RSP0/CP00:Node-D# show bgp l2vpn evpn rd 1.1.1.4:101
[1][0000.0000.0000.0000.0000][12]/120
BGP routing table entry for [1][0000.0000.0000.0000.0000][12]/120, Route Distinguisher: 1.1.1.4:101
Versions:
  Process bRIB/RIB SendTblVer
  Speaker 569 569
Last Modified: Jul 10 18:42:12.455 for 00:45:38
Paths: (1 available, best #1)
  Not advertised to any peer
Path #1: Received by speaker 0
  Not advertised to any peer
Local
  1.1.1.6 C:10000 (bsid:80047) (metric 40) from 1.1.1.253 (1.1.1.6)
  Received Label 80060
  Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported, rib-install
  Received Path ID 0, Local Path ID 1, version 568
  Extended community: Color:10000 RT:65000:101
  Originator: 1.1.1.6, Cluster list: 1.1.1.253
  SR policy color 10000, up, registered, bsid 80047, if-handle 0x00001720
Source AFI: L2VPN EVP, Source VRF: default, Source Route Distinguisher: 1.1.1.6:101

Use the show l2vpn xconnect command to display the state associated with EVPN-VPWS EVI 101 service.

RP/0/RSP0/CP00:Node-D# show l2vpn xconnect group evpn_vpws_group
XConnect Group Name ST Segment 1 Description ST Segment 2 Description ST
------------------------ ----------------------------- -----------------------------
evpn_vpws_group
  evpn_vpws_101          UP           Gi0/0/0/1.2500          UP          EVPN 101,12,1.1.1.6          UP

The following output shows the details for the service. Note that the service is associated with the on-demand SR policy with color 10000 and end-point 1.1.1.6 (node B).
RP/0/RSP0/CPU0:Node-D# show l2vpn xconnect group evpn_vpws_group xc-name evpn_vpws_101

Group evpn_vpws_group, XC evpn_vpws_101, state is up; Interworking none
AC: GigabitEthernet0/0/0/0, state is up
  Type VLAN; Num Ranges: 1
  Rewrite Tags: []
  VLAN ranges: [2500, 2500]
  MTU 1500; XC ID 0x120000c; interworking none
Statistics:
  packets: received 0, sent 0
  bytes: received 0, sent 0
  drops: illegal VLAN 0, illegal length 0
EVPN: neighbor 1.1.1.6, PW ID: evi 101, ac-id 12, state is up (established)
  XC ID 0xa000000d
Encapsulation MPLS
Source address 1.1.1.4
Encap type Ethernet, control word enabled
Sequencing not set
Preferred path Active: SR TE srte_c_10000_ep_1.1.1.6, On-Demand, fallback enabled
Tunnel: Up
Load Balance Hashing: src-dst-mac

<table>
<thead>
<tr>
<th>EVPN</th>
<th>Local</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>80045</td>
<td>80060</td>
</tr>
<tr>
<td>MTU</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Control word enabled</td>
<td>enabled</td>
<td></td>
</tr>
<tr>
<td>AC ID</td>
<td>22</td>
<td>12</td>
</tr>
<tr>
<td>EVPN type</td>
<td>Ethernet</td>
<td>Ethernet</td>
</tr>
</tbody>
</table>

Create time: 10/07/2019 18:42:07 (00:45:49 ago)
Last time status changed: 10/07/2019 18:42:09 (00:45:47 ago)
Statistics:
  packets: received 0, sent 0
  bytes: received 0, sent 0

Use the show segment-routing traffic-eng policy command with tabular option to display SR policy summary information.

The following output shows the on-demand SR policy with BSID 80047 that was triggered by EVPN RT1 prefix with color 10000 advertised by node B (1.1.1.6).
RP/0/RSP0/CPU0:Node-D# show segment-routing traffic-eng policy color 10000 tabular

<table>
<thead>
<tr>
<th>Color</th>
<th>Endpoint</th>
<th>Admin State</th>
<th>Oper State</th>
<th>Binding SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>1.1.1.6</td>
<td>up</td>
<td>up</td>
<td>80047</td>
</tr>
</tbody>
</table>

The following output shows the details for the on-demand SR policy. Note that the SR policy's active candidate path (preference 100) is computed by SR-PCE (1.1.1.207).

Based on the goals of this use case, SR-PCE computes link-disjoint paths for the SR policies associated with a pair of ELINE services between site 1 and site 2. Specifically, from site 2 to site 1, LSP at Node D (srte_c_10000_ep_1.1.1.6) is link-disjoint from LSP at Node C (srte_c_10000_ep_1.1.1.5).
RP/0/RSP0/CPU0:Node-D# show segment-routing traffic-eng policy color 10000

SR-TE policy database
Configure the Head-End Router as PCEP PCC

Configure the head-end router as PCEP Path Computation Client (PCC) to establish a connection to the PCE. The PCC and PCE addresses must be routable so that TCP connection (to exchange PCEP messages) can be established between PCC and PCE.

Configure the PCC to Establish a Connection to the PCE

Use the `segment-routing traffic-eng pcc` command to configure the PCC source address, the SR-PCE address, and SR-PCE options.

A PCE can be given an optional precedence. If a PCC is connected to multiple PCEs, the PCC selects a PCE with the lowest precedence value. If there is a tie, a PCE with the highest IP address is chosen for computing path. The precedence value range is from 0 to 255.

```
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# pcc
Router(config-sr-te-pcc)# source-address ipv4 local-source-address
Router(config-sr-te-pcc)# pce address ipv4 PCE-address[precedence value]
Router(config-sr-te-pcc)# pce address ipv4 PCE-address[password {clear | encrypted} LINE]
Router(config-sr-te-pcc)# pce address ipv4 PCE-address[keychain WORD]
```

Configure PCEP-Related Timers

Use the `timers keepalive` command to specify how often keepalive messages are sent from PCC to its peers. The range is from 0 to 255 seconds; the default value is 30.

```
Router(config-sr-te-pcc)# timers keepalive seconds
```
Use the `timers deadtimer` command to specify how long the remote peers wait before bringing down the PCEP session if no PCEP messages are received from this PCC. The range is from 1 to 255 seconds; the default value is 120.

```
Router(config-sr-te-pcc)# timers deadtimer seconds
```

Use the `timers delegation-timeout` command to specify how long a delegated SR policy can remain up without an active connection to a PCE. The range is from 0 to 3600 seconds; the default value is 60.

```
Router(config-sr-te-pcc)# timers delegation-timeout seconds
```

**PCE-Initiated SR Policy Timers**

Use the `timers initiated orphans` command to specify the amount of time that a PCE-initiated SR policy will remain delegated to a PCE peer that is no longer reachable by the PCC. The range is from 10 to 180 seconds; the default value is 180.

```
Router(config-sr-te-pcc)# timers initiated orphans seconds
```

Use the `timers initiated state` command to specify the amount of time that a PCE-initiated SR policy will remain programmed while not being delegated to any PCE. The range is from 15 to 14440 seconds (24 hours); the default value is 600.

```
Router(config-sr-te-pcc)# timers initiated state seconds
```

To better understand how the PCE-initiated SR policy timers operate, consider the following example:

- PCE A instantiates SR policy P at head-end N.
- Head-end N delegates SR policy P to PCE A and programs it in forwarding.
- If head-end N detects that PCE A is no longer reachable, then head-end N starts the PCE-initiated orphan and state timers for SR policy P.
- If PCE A reconnects before the orphan timer expires, then SR policy P is automatically delegated back to its original PCE (PCE A).
- After the orphan timer expires, SR policy P will be eligible for delegation to any other surviving PCE(s).
- If SR policy P is not delegated to another PCE before the state timer expires, then head-end N will remove SR policy P from its forwarding.

**Enable SR-TE SYSLOG Alarms**

Use the `logging policy status` command to enable SR-TE related SYSLOG alarms.

```
Router(config-sr-te)# logging policy status
```

**Enable PCEP Reports to SR-PCE**

Use the `report-all` command to enable the PCC to report all SR policies in its database to the PCE.

```
Router(config-sr-te-pcc)# report-all
```
Customize MSD Value Signaled by PCC

Use the `maximum-sid-depth value` command to customize the Maximum SID Depth (MSD) signaled by PCC during PCEP session establishment.

The default MSD value is equal to the maximum MSD supported by the platform (10).

```bash
Router(config-sr-te)# maximum-sid-depth value
```

A PCC can signal its MSD to the PCE in the following ways:

- During PCEP session establishment – The signaled MSD is treated as a node-wide property.
  - MSD is configured under `segment-routing traffic-eng maximum-sid-depth value` command
- During PCEP LSP path request – The signaled MSD is treated as an LSP property.
  - On-demand (ODN) SR Policy: MSD is configured using the `segment-routing traffic-eng on-demand color color maximum-sid-depth value` command
  - Local SR Policy: MSD is configured using the `segment-routing traffic-eng policy WORD candidate-paths preference preference dynamic metric sid-limit value` command.

After path computation, the PCE verifies whether the resulting label stack size complies with the MSD requirement. If the label stack size is larger than the MSD, then the PCE returns a "no path" response to the PCC.

Configure PCEP Redundancy Type

Use the `redundancy pcc-centric` command to enable PCC-centric high-availability model, where the PCC allows only the PCE with the lowest precedence to initiate policies.

```bash
Router(config-sr-te-pcc)# redundancy pcc-centric
```

Configuring Head-End Router as PCEP PCC and Customizing SR-TE Related Options: Example

The following example shows how to configure an SR-TE head-end router with the following functionality:

- Enable the SR-TE head-end router as a PCEP client (PCC) with 3 PCEP servers (PCE) with different precedence values. The PCE with IP address 1.1.1.57 is selected as BEST.
- Enable SR-TE related syslogs.
- Set the Maximum SID Depth (MSD) signaled during PCEP session establishment to 5.
- Enable PCEP reporting for all policies in the node.

```bash
segment-routing
traffic-eng
pcc
  source-address ipv4 1.1.1.2
  pce address ipv4 1.1.1.57
  precedence 150
  password clear <password>

! pce address ipv4 1.1.1.58
  precedence 200
  password clear <password>
!`
pce address ipv4 1.1.1.59
prece 250
password clear <password>
!
logging
policy status
!
maximum-sid-depth 5
pcc
report-all
!
!
end

Verification
RP/0/RSP0/CPU0:Router# show segment-routing traffic-eng pcc ipv4 peer

PCC's peer database:
--------------------

Peer address: 1.1.1.57, Precedence: 150, (best PCE)
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 1.1.1.58, Precedence: 200
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 1.1.1.59, Precedence: 250
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

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.
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
RP/0/0/CPU0:xrvr-5(config-expl-path)# index 10 next-address strict ipv4 unicast 192.168.59.9
RP/0/0/CPU0:xrvr-5(config-expl-path)# index 20 next-address strict ipv4 unicast 10.1.1.10
RP/0/0/CPU0:xrvr-5(config-expl-path)# exit
RP/0/0/CPU0:xrvr-5(config)# interface tunnel-te1
RP/0/0/CPU0:xrvr-5(config-if)# ipv4 unnumbered Loopback0
RP/0/0/CPU0:xrvr-5(config-if)# destination 10.1.1.10
RP/0/0/CPU0:xrvr-5(config-if)# path-option 1 explicit name PATH5-9_10 segment-routing
RP/0/0/CPU0:xrvr-5(config-if)# commit
RP/0/0/CPU0:xrvr-5# show mpls traffic-eng tunnels 1 detail
Name: tunnel-te1 Destination: 10.1.1.10 Ifhandle:0x680
Signalled-Name: xrvr-5_t1
Status: Admin: up Oper: up Path: valid Signalling: connected
path option 1, (Segment-Routing) type dynamic (Basis for Setup, path weight 10)
<...>
Binding SID: 24012
<...>
Segment Routing Path Info (IS-IS 1 level-2)
Segment0{Link}: 192.168.59.5 - 192.168.59.9, Label: 24007
Segment1{Node}: 10.1.1.10, Label: 16010
```

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
```
Configure SR-TE Policies

Stitching SR-TE Policies Using Binding SID: Example

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-3(config)# 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_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 PATH4_6_5 (Basis for Setup)
<...>
Binding SID: 24008
<...>
Segment-Routing Path Info (IS-IS 1 level-2)
  Segment0[Node]: 10.1.1.4, Label: 16004
  Segment1[Link]: 192.168.46.4 - 192.168.46.6, Label: 24003
  Segment2[Node]: 10.1.1.5, Label: 16005
  Segment3[ - ]: Label: 24012
```

```
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
```

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x
<table>
<thead>
<tr>
<th></th>
<th>IP Address</th>
<th>MPLS Label</th>
<th>Exp</th>
<th>In 1</th>
<th>In 2</th>
<th>Out 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>99.2.3.3</td>
<td>24008</td>
<td>0</td>
<td>29</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>99.3.4.4</td>
<td>24003/16005/24012</td>
<td>0</td>
<td>29</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>4</td>
<td>99.4.6.6</td>
<td>16005/24012</td>
<td>0</td>
<td>29</td>
<td>29</td>
<td>19</td>
</tr>
<tr>
<td>5</td>
<td>99.5.6.5</td>
<td>24012</td>
<td>0</td>
<td>29</td>
<td>29</td>
<td>19</td>
</tr>
<tr>
<td>6</td>
<td>99.5.9.9</td>
<td>16010</td>
<td>0</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>7</td>
<td>99.9.10.10</td>
<td>19</td>
<td></td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
</tbody>
</table>
Enabling Segment Routing Flexible Algorithm

Segment Routing (SR) allows a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called segments. Segment Routing also defines an algorithm that dictates how the path is computed across the network. It also provides a way to associate prefix-SID with an algorithm. This allows IGPs to compute the path based on various algorithms and forward the traffic on such path using the algorithm specific segments.

This document describes the IS-IS extension to support Segment Routing Flexible Algorithm on an MPLS data-plane.

- Prerequisites for Flexible Algorithm, on page 121
- Building Blocks of Segment Routing Flexible Algorithm, on page 121
- Configuring Flexible Algorithm, on page 123
- Example: Configuring IS-IS Flexible Algorithm, on page 124
- Example: Traffic Steering to Flexible Algorithm Paths, on page 125

Prerequisites for Flexible Algorithm

Segment routing must be enabled on the router before the Flexible Algorithm functionality is activated.

Building Blocks of Segment Routing Flexible Algorithm

This section describes the building blocks that are required to support the SR Flexible Algorithm functionality in IS-IS.

Flexible Algorithm Definition

Many possible constrains may be used to compute a path over a network. Some networks are deployed with multiple planes. A simple form of constrain may be to use a particular plane. A more sophisticated form of constrain can include some extended metric, like delay, as described in [RFC7810]. Even more advanced case could be to restrict the path and avoid links with certain affinities. Combinations of these are also possible. To provide a maximum flexibility, the mapping between the algorithm value and its meaning can be defined by the user. When all the routers in the domain have the common understanding what the particular algorithm value represents, the computation for such algorithm is consistent and the traffic is not subject to looping. Here, since the meaning of the algorithm is not defined by any standard, but is defined by the user, it is called as Flexible Algorithm.
Flexible Algorithm Support Advertisement

An algorithm defines how the best path is computed by IGP. Routers advertise the support for the algorithm as a node capability. Prefix-SIDs are also advertised with an algorithm value and are tightly coupled with the algorithm itself.

An algorithm is a one octet value. Values from 128 to 255 are reserved for user defined values and are used for Flexible Algorithm representation.

Flexible Algorithm Definition Advertisement

To guarantee the loop free forwarding for paths computed for a particular Flexible Algorithm, all routers in the network must share the same definition of the Flexible Algorithm. This is achieved by dedicated router(s) advertising the definition of each Flexible Algorithm. Such advertisement is associated with the priority to make sure that all routers will agree on a single and consistent definition for each Flexible Algorithm.

Definition of Flexible Algorithm includes:

- Metric type
- Affinity constraints

To enable the router to advertise the definition for the particular Flexible Algorithm, advertise-definition command is used. At least one router in the area, preferably two for redundancy, must advertise the Flexible Algorithm definition. Without the valid definition being advertised, the Flexible Algorithm will not be functional.

Flexible Algorithm Prefix-SID Advertisement

To be able to forward traffic on a Flexible Algorithm specific path, all routers participating in the Flexible Algorithm will install an MPLS labeled path for the Flexible Algorithm specific SID that is advertised for the prefix. Only prefixes for which the Flexible Algorithm specific Prefix-SID is advertised is subject to Flexible Algorithm specific forwarding.

Calculation of Flexible Algorithm Path

A router may compute path for multiple Flexible Algorithms. A router must be configured to support particular Flexible Algorithm before it can compute any path for such Flexible Algorithm. A router must have a valid definition of the Flexible Algorithm before such Flexible Algorithm is used.

When computing the shortest path tree for particular Flexible Algorithm:

- All nodes that do not advertise support for such Flexible Algorithm will be pruned from the topology.
- If the Flexible Algorithm definition includes affinities that are excluded, then all links for which any of such affinities are advertised will be pruned from the topology.
- Router uses the metric that is part of the Flexible Algorithm definition. If the metric is not advertised for the particular link, such link will be pruned from the topology.

Loop Free Alternate (LFA) paths for particular Flexible Algorithm path are computed using the same constrains as the calculation of the primary paths for such Flexible Algorithm. LFA paths only use the Prefix-SIDs advertised specifically for such Flexible Algorithm to enforce the traffic over such path.
Installation of Forwarding Entries for Flexible Algorithm Paths

Flexible Algorithm path to any prefix must be installed in the forwarding using the Prefix-SID that was advertised for such Flexible Algorithm. If the Prefix-SID for Flexible Algorithm is not known, such Flexible Algorithm path is not installed in forwarding for such prefix.

Only MPLS to MPLS entries are installed for a Flexible Algorithm path. No IP to IP or IP to MPLS entries are installed. These follow the native IPG paths computed based on the default algorithm and regular IGP metrics.

Configuring Flexible Algorithm

For information about the commands usage, see the Segment Routing Command Reference for Cisco ASR 9000 Series Routers.

The following ISIS configuration sub-mode is used to configure Flexible Algorithm:

```
flex-algo algorithm number
```

`algorithm number` — value from 128 to 255

**Commands under Flexible Algorithm Configuration Mode**

The following commands are used to configure Flexible Algorithm definition under the flex-algo sub-mode:

* `metric-type delay`

**Note**

By default the regular IGP metric is used. If delay metric is enabled, the advertised delay on the link is used as a metric for Flexible Algorithm computation.

* `affinity exclude-any name1, name2, ...`

`name` — name of the affinity map

* `priority priority value`

`priority value` — priority used during the Flexible Algorithm definition election.

The following command is used to enable advertisement of the Flexible Algorithm definition in IS-IS:

```
advertise-definition
```

**Commands for Affinity Configuration**

The following command is used for defining the affinity-map. Affinity-map associates the name with the particular bit positions in the Extended Admin Group bitmask.

```
affinity-map name bit-position bit number
```
name—name of the affinity-map.

bit number—bit position in the Extended Admin Group bitmask.

The following command is used to associate the affinity with an interface:

```
affinity flex-algo name 1, name 2, ...
```

name—name of the affinity-map

Command for Prefix-SID Configuration

The following command is used to advertise prefix-SID for default and strict-SPF algorithm:

```
prefix-sid [strict-spf | algorithm algorithm-number] [index | absolute] sid value
```

- algorithm-number—Flexible Algorithm number
- sid value—SID value

Example: Configuring IS-IS Flexible Algorithm

```
router isis 1
  affinity-map red bit-position 65
  affinity-map blue bit-position 8
  affinity-map green bit-position 201

  flex-algo 128
  advertise-definition
  exclude affinity red blue

  flex-algo 129
  exclude affinity green

  address family ipv4 unicast
  segment-routing mpls

  interface Loopback0
  address-family ipv4 unicast
  prefix-sid algorithm 128 index 100
  prefix-sid algorithm 129 index 101

  interface GigabitEthernet0/0/0/0
  affinity flex-algo red

  interface GigabitEthernet0/0/0/1
  affinity flex-algo blue red

  interface GigabitEthernet0/0/0/2
  affinity flex-algo blue
```

Enabling Segment Routing Flexible Algorithm

Example: Configuring IS-IS Flexible Algorithm
Example: Traffic Steering to Flexible Algorithm Paths

BGP Routes on PE – Color Based Steering

SR-TE On Demand Next-Hop (ODN) feature can be used to steer the BGP traffic towards the Flexible Algorithm paths.

The following example configuration shows how to setup BGP steering local policy, assuming two router: R1 (2.2.2.2) and R2 (4.4.4.4), in the topology.

Configuration on router R1:

```
vrf Test
address-family ipv4 unicast
  import route-target
  1:150
  !
  export route-policy SET_COLOR_RED_HI_BW
  export route-target
  1:150
  !

interface Loopback0
  ipv4 address 2.2.2.2 255.255.255.255
!
interface Loopback150
  vrf Test
  ipv4 address 2.2.2.222 255.255.255.255
!
interface TenGigE0/1/0/3/0
  description exr1 to cxr1
  ipv4 address 10.0.20.2 255.255.255.0
  !
  extcommunity-set opaque color129-red-igp 129
end-set
!
route-policy PASS
  pass
end-policy
!
route-policy SET_COLOR_RED_HI_BW
  set extcommunity color color129-red-igp 129
  pass
end-policy
!
router isis 1
  is-type level-2-only
  net 49.0001.0000.0000.0002.00
  log adjacency changes
  affinity-map RED bit-position 28
  flex-algo 128
  priority 228
!
address-family ipv4 unicast
  metric-style wide
  advertise link attributes
  router-id 2.2.2.2
  segment-routing mpls
```
interface Loopback0
  address-family ipv4 unicast
  prefix-sid index 2
  prefix-sid algorithm 128 index 282

interface TenGigE0/1/0/3/0
  point-to-point
  address-family ipv4 unicast

router bgp 65000
  bgp router-id 2.2.2.2
  address-family ipv4 unicast

  address-family vpnv4 unicast
  retain route-target all

neighbor-group RR-services-group
  remote-as 65000
  update-source Loopback0
  address-family ipv4 unicast
  address-family vpnv4 unicast

neighbor 4.4.4.4
  use neighbor-group RR-services-group

vrf Test
  rd auto
  address-family ipv4 unicast
  redistribute connected

segment-routing
  traffic-eng
  logging
  policy status

  segment-list sl-cxr1
  index 10 mpls label 16294

  policy pol-foo
  color 129 end-point ipv4 4.4.4.4
  candidate-paths
  preference 100
  explicit segment-list sl-cxr1

Configuration on router R2:

vrf Test
  address-family ipv4 unicast
  import route-target
  1:150

  export route-policy SET_COLOR_RED_HI_BW
export route-target
  1:150

interface TenGigE0/1/0/1
description cxr1 to exr1
ipv4 address 10.0.20.1 255.255.255.0
extcommunity-set opaque color129-red-igp 129
end-set
route-policy PASS
pass
end-policy
route-policy SET_COLOR_RED_HI_BW
  set extcommunity color color129-red-igp
  pass
end-policy
router isis 1
  is-type level-2-only
  net 49.0001.0000.0000.0004.00
  log adjacency changes
  affinity-map RED bit-position 28
  affinity-map BLUE bit-position 29
  affinity-map GREEN bit-position 30
  flex-algo 128
    priority 228
  flex-algo 129
    priority 229
  flex-algo 130
    priority 230
  address-family ipv4 unicast
    metric-style wide
    advertise link attributes
    router-id 4.4.4.4
    segment-routing mpls
interface Loopback0
  address-family ipv4 unicast
    prefix-sid index 4
    prefix-sid algorithm 128 index 284
    prefix-sid algorithm 129 index 294
    prefix-sid algorithm 130 index 304
interface GigabitEthernet0/0/0/0
  point-to-point
  address-family ipv4 unicast
interface TenGigE0/1/0/1
  point-to-point
  address-family ipv4 unicast
router bgp 65000
  bgp router-id 4.4.4.4
address-family ipv4 unicast
!
address-family vpnv4 unicast
!
neighbor-group RR-services-group
  remote-as 65000
  update-source Loopback0
  address-family ipv4 unicast
  !
  address-family vpnv4 unicast
  !
!
neighbor 1.1.1.1
  use neighbor-group RR-services-group
!
neighbor 2.2.2.2
  use neighbor-group RR-services-group
!
vrf Test
  rd auto
  address-family ipv4 unicast
    redistribute connected
  !
neighbor 25.1.1.2
  remote-as 4
  address-family ipv4 unicast
    route-policy PASS in
    route-policy PASS out
  !
!
!
segment-routing
!
end
CHAPTER 10

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 129
• Configure SR-PCE, on page 130
• PCE-Initiated SR Policies for Traffic Management, on page 133
• ACL Support for PCEP Connection, on page 135

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> configure</td>
<td>Enables PCE and enters PCE configuration mode.</td>
</tr>
<tr>
<td><strong>Step 2</strong> pce</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
</tbody>
</table>
| RP/0/RSP0/CPU0:router(config)# pce | **Purpose**

**Step 3**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>address ipv4 <em>address</em></td>
<td>Configures a PCE IPv4 address.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# address ipv4 192.168.0.1</td>
<td>Configures the remote peer for state synchronization.</td>
</tr>
</tbody>
</table>

**Step 4**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>state-sync ipv4 <em>address</em></td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# state-sync ipv4 192.168.0.3</td>
<td>Configures the remote peer for state synchronization.</td>
</tr>
</tbody>
</table>

**Step 5**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp-buffer <em>size</em></td>
<td>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.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# tcp-buffer 1024000</td>
<td>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.</td>
</tr>
</tbody>
</table>

**Step 6**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>password (clear</td>
<td>encrypted) <em>password</em></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# password encrypted pwd1</td>
<td>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.</td>
</tr>
</tbody>
</table>

**Step 7**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>segment-routing {strict-sid-only</td>
<td>te-latency}</td>
</tr>
<tr>
<td>Example:</td>
<td>This setting is global and applies to all LSPs that request a path from this controller.</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# segment-routing strict-sid-only</td>
<td></td>
</tr>
</tbody>
</table>

**Step 8**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>timers</td>
<td>Enters timer configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce)# timers</td>
<td></td>
</tr>
</tbody>
</table>

**Step 9**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>keepalive <em>time</em></td>
<td>Configures the timer value for locally generated keep-alive messages. The default time is 30 seconds.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-pce-timers)# keepalive 60</td>
<td></td>
</tr>
</tbody>
</table>

Segment Routing Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.6.x 131
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>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
</tbody>
</table>
| RP/0/RSP0/CPU0:router(config-pce-disjoint)# group-id 1 type node sub-id 1 | - **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, 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
**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
**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
RP/0/RSP0/CPU0:router(config-pce-disjoint)# lsp 2 pcc_ipv4 192.168.0.5 lsp-name rtrE_t2
```

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.
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 63.

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.

```bash
/* 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

```plaintext
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
!
```

ACL Support for PCEP Connection

PCE protocol (PCEP) (RFC5440) is a client-server model running over TCP/IP, where the server (PCE) opens a port and the clients (PCC) initiate connections. After the peers establish a TCP connection, they create a PCE session on top of it.

The ACL Support for PCEP Connection feature provides a way to protect a PCE server using an Access Control List (ACL) to restrict IPv4 PCC peers at the time the TCP connection is created based on the source address of a client. When a client initiates the TCP connection, the ACL is referenced, and the client source address is compared. The ACL can either permit or deny the address and the TCP connection will proceed or not.

Refer to the Implementing Access Lists and Prefix Lists chapter in the *IP Addresses and Services Configuration Guide for Cisco ASR 9000 Series Routers* for detailed ACL configuration information.

To apply an ACL to the PCE, use the `pce peer-filter ipv4 access-list acl_name` command.
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 137
- Configuring TI-LFA for OSPF, on page 139
- TI-LFA Node and SRLG Protection: Examples, on page 141
- Configuring Global Weighted SRLG Protection, on page 142

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>configure</td>
<td>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<td>Step 2</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>Example:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# router isis 1</td>
<td>Note: 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>Step 3</td>
<td>interface type interface-path-id</td>
<td>Enters interface configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-isis)# interface GigabitEthernet0/0/2/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-isis)# interface Bundle-Ether1</td>
<td></td>
</tr>
<tr>
<td>Step 4</td>
<td>address-family ipv4 [unicast]</td>
<td>Specifies the IPv4 address family, and enters router address family configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast</td>
<td></td>
</tr>
<tr>
<td>Step 5</td>
<td>fast-reroute per-prefix</td>
<td>Enables per-prefix fast reroute.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 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

TI-LFA has been successfully configured for segment routing.
### 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 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** | area *area-id*  
Example:  
RP/0/RSP0/CPU0:router(config-ospf)# area 1 | Enters area configuration mode. |
| **Step 4** | interface *type* *interface-path-id*  
Example:  
RP/0/RSP0/CPU0:router(config-ospf-ar)# interface GigabitEthernet0/0/2/1 | Enters interface configuration mode. |
| **Step 5** | fast-reroute per-prefix  
Example:  
RP/0/RSP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix | Enables per-prefix fast reroute. |
| **Step 6** | fast-reroute per-prefix ti-lfa  
Example:  
RP/0/RSP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix ti-lfa | Enables per-prefix TI-LFA fast reroute link protection. |
| **Step 7** | fast-reroute per-prefix tiebreaker {node-protecting | srlg-disjoint} index *priority*  
Example:  
RP/0/RSP0/CPU0:router(config-isis-ar-if)# fast-reroute per-prefix srlg-disjoint index 100 | 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.  
**Note** 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.

```plaintext
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-if)# 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
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)# 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)# interface TenGigE0/0/0
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)# interface TenGigE0/0/1
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
```

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
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000
```

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.
CHAPTER 12

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 145
- Segment Routing Microloop Avoidance Limitations, on page 145
- Configure Segment Routing Microloop Avoidance for IS-IS, on page 145
- Configure Segment Routing Microloop Avoidance for OSPF, on page 146

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.

Segment Routing Microloop Avoidance Limitations

For IS-IS, Segment Routing Microloop Avoidance is not supported when incremental shortest path first (ISPF) is configured.

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>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 <code>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</code> router configuration command.</td>
</tr>
<tr>
<td>Example: <code>RP/0/RSP0/CPU0:router(config)# router isis 1</code></td>
<td></td>
</tr>
<tr>
<td>Step 3 address-family ipv4 [ unicast ]</td>
<td>Specifies the IPv4 address family and enters router address family configuration mode.</td>
</tr>
<tr>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-isis)# address-family ipv4 unicast</code></td>
<td></td>
</tr>
<tr>
<td>Step 4 microloop avoidance segment-routing</td>
<td>Enables Segment Routing Microloop Avoidance.</td>
</tr>
<tr>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-isis-af)# microloop avoidance segment-routing</code></td>
<td></td>
</tr>
<tr>
<td>Step 5 microloop avoidance rib-update-delay <code>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>Example: <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>configure</td>
<td>Enables OSPF routing for the specified routing process, and places the router in router configuration mode.</td>
</tr>
<tr>
<td>Step 2</td>
<td>router ospf process-name</td>
<td>Enables Segment Routing Microloop Avoidance.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# router ospf 1</td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td>microloop avoidance segment-routing</td>
<td>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.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# microloop avoidance segment-routing</td>
<td></td>
</tr>
<tr>
<td>Step 4</td>
<td>microloop avoidance rib-update-delay delay-time</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# microloop avoidance rib-update-delay 3000</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 13

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 149
- Segment Routing and LDP Interoperability, on page 150
- Configuring Mapping Server, on page 152
- Enable Mapping Advertisement, on page 154
- Enable Mapping Client, on page 156

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 range`
7. `commit`
8. `show segment-routing mapping-server prefix-sid-map [ipv4 | ipv6] [detail]`

### 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>
<tr>
<td>Step 2</td>
<td><code>segment-routing</code></td>
<td>Enables segment routing.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td><code>segment-routing</code></td>
</tr>
<tr>
<td>Step 3</td>
<td><code>mapping-server</code></td>
<td>Enables mapping server configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-sr)#</td>
<td><code>mapping-server</code></td>
</tr>
<tr>
<td>Step 4</td>
<td><code>prefix-sid-map</code></td>
<td>Enables prefix-SID mapping configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Note</td>
<td>Two-way prefix SID can be enabled directly under IS-IS or through a mapping server.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-sr-ms)#</td>
<td><code>prefix-sid-map</code></td>
</tr>
<tr>
<td>Step 5</td>
<td>`address-family ipv4</td>
<td>ipv6`</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This example shows the address-family for ipv4:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-sr-ms-map)#</td>
<td><code>address-family ipv4</code></td>
</tr>
<tr>
<td></td>
<td>This example shows the address-family for ipv6:</td>
<td></td>
</tr>
</tbody>
</table>
### Configure Segment Routing Mapping Server

#### Configuring Mapping Server

**Command or Action**

```
RP/0/RSP0/CPU0:router(config-sr-ms-map)#
address-family ipv6
```

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| 6    | `ip-address/prefix-length first-SID-value range range` | Adds SID-mapping entries in the active local mapping policy. In the configured example:  
- 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  
- 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. |
| 7    | `commit`         | Display information about the locally configured prefix-to-SID mappings. |
| 8    | `show segment-routing mapping-server prefix-sid-map [ipv4 | ipv6] [detail]` | Displays information about the locally configured prefix-to-SID mappings. **Note** Specify the address family for IS-IS. |

**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
```

**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: 400
  Range: 300
  Last Prefix: 20.2.44.0/24
  Last SID Index: 699
  Flags:

Prefix SID Index: 10
  Range: 200
  Last Prefix: 10.1.1.200/32
  Last SID Index: 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> <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><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config)# router isis 1</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> `address-family { ipv4</td>
<td>ipv6 } [ unicast ]`</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-isis)# address-family ipv4 unicast</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>segment-routing prefix-sid-map advertise-local</code></td>
<td>Configures IS-IS to advertise locally configured prefix-SID mappings.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-isis-af)# segment-routing prefix-sid-map advertise-local</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>commit</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> <code>show isis database verbose</code></td>
<td>Displays IS-IS prefix-SID mapping advertisement and TLV.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router# show isis database verbose&lt;br&gt;&lt;...removed...&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>
**Configure Mapping 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>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>router ospf process-name</code></td>
<td>Enables OSPF routing for the specified routing instance, and places the router in router configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# <code>router ospf 1</code></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><code>segment-routing prefix-sid-map advertise-local</code></td>
<td>Configures OSPF to advertise locally configured prefix-SID mappings.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# <code>segment-routing prefix-sid-map advertise-local</code></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><code>commit</code></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>show ospf database opaque-area</code></td>
<td>Displays OSPF prefix-SID mapping advertisement and TLV.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# <code>show ospf database opaque-area</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;...removed...&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Extended Prefix Range TLV:</strong> Length: 24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AF: 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prefix: 10.1.1.1/32</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Range Size: 200</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Flags: 0x0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SID sub-TLV: Length: 8</td>
<td></td>
</tr>
</tbody>
</table>
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
```
CHAPTER 14

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 157
- Traffic Collector Process, on page 157
- Configuring Traffic Collector, on page 158
- Displaying Traffic Information, on page 159

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>1. configure</td>
<td>Enables traffic collector and places the router in traffic collector configuration mode.</td>
</tr>
<tr>
<td>2. <code>traffic-collector</code></td>
<td>Enables traffic collector and places the router in traffic collector configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RP0/CPU0:router(config)# <code>traffic-collector</code></td>
</tr>
<tr>
<td>3. <code>statistics collection-interval value</code></td>
<td>(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.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RP0/CPU0:router(config-tc)# <code>statistics collection-interval 5</code></td>
</tr>
<tr>
<td>4. <code>statistics history-size value</code></td>
<td>(Optional) Specifies the number of entries kept in the history database. Valid values are from 1 to 10. The default is 5.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
</tbody>
</table>
### Displaying Traffic Information

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

#### Note
For detailed information about the command syntax for the following `show` commands, see the *Segment Routing Command Reference Guide*.

- **Display the configured external interfaces:**

  ```
  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:**

  ```
  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
  ```

This completes the configuration for the traffic collector.
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 of the Base and TM 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 161
- Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID, on page 162
- MPLS LSP Ping and Traceroute Nil-FEC Target, on page 164
- Examples: LSP Ping and Traceroute for Nil_FEC Target, on page 164
- Segment Routing Ping, on page 165
- Segment Routing Traceroute, on page 168
- Segment Routing Policy Nil-FEC Ping and Traceroute, on page 170
- Segment Routing over IPv6 OAM, on page 172

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:

```
R1 -- 1.1.1.1 -- R2
     |       |
     |       |
     V       V
R5 -- 1.1.1.4 -- R6

R3

R4

1.1.1.4
```

### MPLS Ping for Prefix-SID

```
RP/0/RSP0/CPU0:router-arizona# 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

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 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

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.

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 5: 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} [nexthop nexthop-ip-addr]]</td>
</tr>
<tr>
<td>traceroute mpls nil-fec labels {label[ , label]} [output {interface tx-interface} [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</th>
<th>Outgoing Prefix</th>
<th>Outgoing Interface</th>
<th>Next Hop</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>Label or ID</td>
<td>Switched</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----------------</td>
<td>--------------------</td>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td>16004</td>
<td>Pop No ID</td>
<td>G10/2/0/0/1</td>
<td>10.1.1.4</td>
<td>1392</td>
</tr>
<tr>
<td>Pop</td>
<td>No ID</td>
<td>G10/2/0/0/2</td>
<td>10.1.2.2</td>
<td>0</td>
</tr>
<tr>
<td>16005</td>
<td>16005 No ID</td>
<td>G10/2/0/0/0</td>
<td>10.1.1.4</td>
<td>0</td>
</tr>
<tr>
<td>16005</td>
<td>No ID</td>
<td>G10/2/0/0/1</td>
<td>10.1.2.2</td>
<td>0</td>
</tr>
<tr>
<td>16007</td>
<td>16007 No ID</td>
<td>G10/2/0/0/0</td>
<td>10.1.1.4</td>
<td>4752</td>
</tr>
<tr>
<td>16007</td>
<td>No ID</td>
<td>G10/2/0/0/1</td>
<td>10.1.2.2</td>
<td>0</td>
</tr>
<tr>
<td>24000</td>
<td>Pop SR Adj (idx 0)</td>
<td>G10/2/0/0/0</td>
<td>10.1.1.4</td>
<td>0</td>
</tr>
<tr>
<td>24001</td>
<td>Pop SR Adj (idx 2)</td>
<td>G10/2/0/0/0</td>
<td>10.1.1.4</td>
<td>0</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:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'!' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'l' - Label switched with FEC change, 'd' - see DDMAP for return code,
'X' - unknown return code, 'x' - return code 0

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

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

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'L' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'l' - Label switched with FEC change, 'd' - see DDMAP for return code,
'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

0 10.1.1.3 MRU 1500 [Labels: 16005/16007/explicit-null Exp: 0/0/0]
L 1 10.1.1.4 MRU 1500 [Labels: implicit-null/16007/explicit-null Exp: 0/0/0] 1 ms
L 2 10.1.1.5 MRU 1500 [Labels: implicit-null/explicit-null Exp: 0/0/0] 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.

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.

```
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
```

```
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 traceraceroute 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

Type escape sequence to abort.

0 10.12.12.1 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.12.12.2 2 ms

RP/0/RSP0/CP0:router# traceroute sr-mpls 10.1.1.2/32 fec-type igp ospf

Tracing MPLS Label Switched Path to 10.1.1.2/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'L' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

0 10.12.12.1 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.12.12.2 2 ms

RP/0/RSP0/CP0:router# traceroute sr-mpls 10.1.1.2/32 fec-type igp isis

Tracing MPLS Label Switched Path to 10.1.1.2/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'L' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

0 10.12.12.1 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 10.12.12.2 2 ms

RP/0/RSP0/CP0:router# traceroute sr-mpls 10.1.1.2/32 fec-type bgp

Tracing MPLS Label Switched Path to 10.1.1.2/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'L' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

0 10.12.12.1 MRU 1500 [Labels: implicit-null/implicit-null Exp: 0/0]
! 1 10.12.12.2 2 ms

This example shows how to use multipath traceroute to discover all the possible paths for a IPv4 prefix SID.

RP/0/RSP0/CP0:router# traceroute sr-mpls multipath 10.1.1.2/32
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:
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/CPU0: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 99.1.2.1 MRU 4470 [Labels: 16002/16004/explicit-null Exp: 0/0/0] 3 ms
L 1 99.1.2.2 MRU 4470 [Labels: implicit-null/explicit-null Exp: 0/0] 4 ms
```

```
RP/0/0/CPU0:router# traceroute sr-mpls nil-fec binding-sid 100001
Tracing MPLS Label Switched Path with Nil FEC with labels [16002/16004], 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 99.1.2.1 MRU 4470 [Labels: 16002/16004/explicit-null Exp: 0/0/0] 3 ms
L 1 99.1.2.2 MRU 4470 [Labels: 16004/explicit-null Exp: 0/0] 3 ms
L 2 99.2.6.6 MRU 4470 [Labels: implicit-null Exp: 0] 3 ms
! 3 99.4.6.4 11 ms
```
Segment Routing over IPv6 OAM

Segment Routing over IPv6 data plane (SRv6) implementation adds a new type of routing extension header. Hence, the existing ICMPv6 mechanisms including ping and traceroute can be used in the SRv6 network. There is no change in the way ping and traceroute operations work for IPv6- or SRv6-capable nodes in an SRv6 network.

Restrictions and Usage Guidelines

The following restriction applies for SRv6 OAM:

- Ping to an SRv6 SID is not supported.

Examples: SRv6 OAM

The following example shows using ping in an SRv6 network.

```
RP/0/RP0/CPU0:Router# ping ipv6 2001::33:33:33:33
Mon Sep 17 20:04:10.068 UTC
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001::33:33:33:33, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
```

The following example shows using traceroute in an SRv6 network.

```
RP/0/RP0/CPU0:Router# traceroute ipv6 2001::33:33:33 probe 1 timeout 0 srv6
Fri Sep 14 15:59:25.170 UTC
Type escape sequence to abort.
Tracing the route to 2001::33:33:33:33
  3 2001::44:44:44:44 2 msec
  4 2001::33:33:33:33 3 msec
```

The following example shows using traceroute in an SRv6 network without an SRH.

```
RP/0/RSP1/CPU0:Router# traceroute ipv6 2001::44:44:44:44 srv6
Wed Jan 16 14:35:27.511 UTC
Type escape sequence to abort.
Tracing the route to 2001::44:44:44:44
  1 2001::2:2:2:2 3 msec 2 msec 2 msec
  2 2001::44:44:44:44 3 msec 3 msec 3 msec
```

The following example shows using ping for a specified IP address in the VRF.

```
RP/0/RP0/CPU0:Router# ping 10.15.15.1 vrf red
Mon Sep 17 20:07:10.085 UTC
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.15.15.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

The following example shows using traceroute for a specified IP address in the VRF.
RP/0/RP0/CPU0:Router# traceroute 10.15.15.1 vrf red
Mon Sep 17 20:07:18.478 UTC

Type escape sequence to abort.
Tracing the route to 10.15.15.1
1  10.15.15.1 3 msec 2 msec 2 msec

The following example shows using traceroute for CE1 (4.4.4.5) to CE2 (5.5.5.5) in the VRF:

RP/0/RP0/CPU0:Router# traceroute 5.5.5.5 vrf a
Wed Jan 16 15:08:46.264 UTC

Type escape sequence to abort.
Tracing the route to 5.5.5.5
1  14.14.14.1 5 msec 1 msec 1 msec
2  15.15.15.1 3 msec 2 msec 2 msec
3  15.15.15.2 2 msec * 3 msec