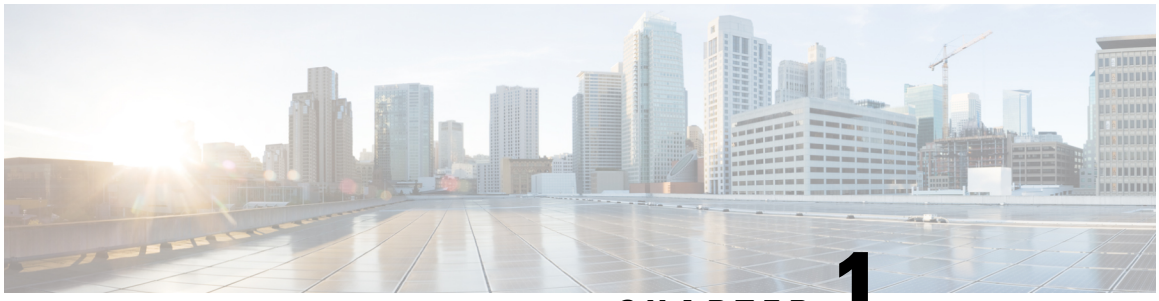
# Segment Routing Configuration Guide for Cisco NCS 540 Series Routers, IOS XR Release 7.6.x

**First Published:** 2022-03-30

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

# CONTENTS



CHAPTER 1

# About Segment Routing



## 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.

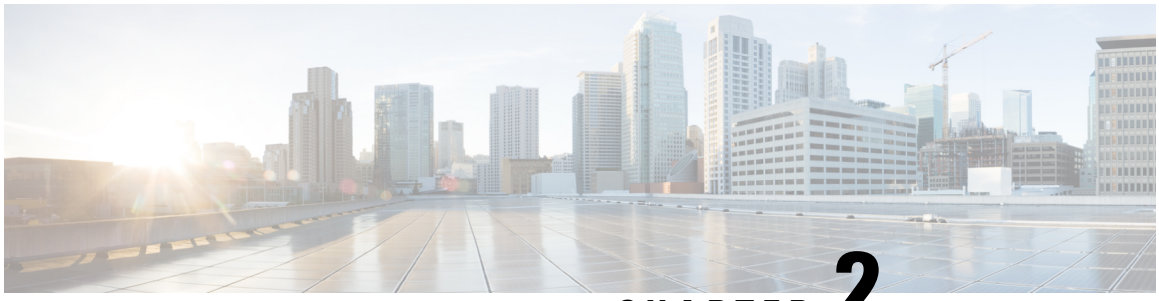
# Benefits

- **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 for the IGP
3. Configure Segment Routing for BGP
4. Configure the SR-TE Policy
5. Configure the SR-PCE
6. Configure TI-LFA and Microloop Avoidance
7. Configure the Segment Routing Mapping Server



CHAPTER 2

# 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 Overview

***Table 1: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SRv6 Network Instructions | Release 7.5.1 | This feature is now supported on Cisco NCS 540 series routers and operate in the native mode. |

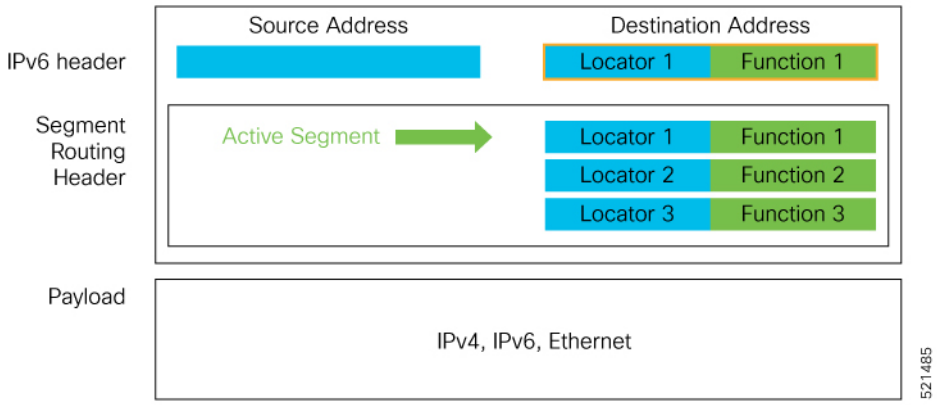
Segment Routing (SR) can be applied on both MPLS and IPv6 data planes. Segment Routing over IPv6 (SRv6) extends Segment Routing support with IPv6 data plane.

In an SR-MPLS enabled network, an MPLS label represents an instruction. The source nodes programs the path to a destination in the packet header as a stack of labels.

SRv6 introduces the Network Programming framework that enables a network operator or an application to specify a packet processing program by encoding a sequence of instructions in the IPv6 packet header. Each instruction is implemented on one or several nodes in the network and identified by an SRv6 Segment Identifier (SID) in the packet. The SRv6 Network Programming framework is defined in IETF RFC 8986 SRv6 Network Programming.

In SRv6, an IPv6 address represents an instruction. SRv6 uses a new type of IPv6 Routing Extension Header, called the Segment Routing Header (SRH), in order to encode an ordered list of instructions. The active segment is indicated by the destination address of the packet, and the next segment is indicated by a pointer in the SRH.

***Figure 1: Network Program in the Packet Header***




The SRv6 SRH is documented in IETF RFC IPv6 Segment Routing Header (SRH).

The SRH is defined as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Last Entry   |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[0] (128-bit IPv6 address)             |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
                               ...
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[n] (128-bit IPv6 address)             |
```

```
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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 *n*th 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.

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.

### SRv6 Node Roles

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.

- Endpoint 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

### SRv6 Head-End Behaviors

The SR Headend with Encapsulation behaviors are documented in the IETF RFC 8986 SRv6 Network Programming.

The SR Headend with Insertion head-end behaviors are documented in the following IETF draft:

https://datatracker.ietf.org/doc/draft-filsfils-spring-srv6-net-pgm-insertion/

This section describes a set of SR Policy headend behaviors. The following list summarizes them:

- H.Encaps—SR Headend Behavior with Encapsulation in an SRv6 Policy
- H.Encaps.Red—H.Encaps with Reduced Encapsulation
- H.Insert—SR Headend with insertion of an SRv6 Policy
- H.Insert.Red—H.Insert with reduced insertion

### SRv6 Endpoint Behaviors

The SRv6 endpoint behaviors are documented in the IETF RFC 8986 SRv6 Network Programming.

The following is a subset of defined SRv6 endpoint behaviors that can be associated with a SID.

- End—Endpoint function. The SRv6 instantiation of a Prefix SID [RFC8402].
- End.X—Endpoint with Layer-3 cross-connect. The SRv6 instantiation of an Adj SID [RFC8402].
- End.DX6—Endpoint with decapsulation and IPv6 cross-connect (IPv6-L3VPN - equivalent to per-CE VPN label).
- End.DX4—Endpoint with decapsulation and IPv4 cross-connect (IPv4-L3VPN - equivalent to per-CE VPN label).
- End.DT6—Endpoint with decapsulation and IPv6 table lookup (IPv6-L3VPN - equivalent to per-VRF VPN label).
- End.DT4—Endpoint with decapsulation and IPv4 table lookup (IPv4-L3VPN - equivalent to per-VRF VPN label).
- End.DT46—Endpoint with decapsulation and specific IP table lookup (IP-L3VPN - equivalent to per-VRF VPN label).
- End.DX2—Endpoint with decapsulation and L2 cross-connect (L2VPN use-case).
- End.B6.Encaps—Endpoint bound to an SRv6 policy with encapsulation. SRv6 instantiation of a Binding SID.
- End.B6.Encaps.RED—End.B6.Encaps with reduced SRH. SRv6 instantiation of a Binding SID.

### SRv6 Endpoint Behavior Variants

*Table 2: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SRv6: Ultimate Segment Decapsulation (USD) on Full-length SIDs | Release 7.5.2 | The Ultimate Segment Decapsulation (USD) variant is supported on SRv6 endpoint nodes using full-length SIDs. One of the USD variant applications is the case of TI-LFA in P routers with encapsulation with H.Encaps. The USD variant allows the last Segment Endpoint Node in the repair path list to decapsulate the IPv6 header added at the TI-LFA Point of Local Repair and forward the inner packet.<br><br>In earlier releases, the USD variant was supported on SRv6 endpoint nodes using Micro SIDs (uSIDs). |

Depending on how the SRH is handled, different behavior variants are defined for the End and End.X behaviors. The End and End.X behaviors can support these variants, either individually or in combinations.

- **Penultimate Segment Pop (PSP) of the SRH variant**—An SR Segment Endpoint Nodes receive the IPv6 packet with the Destination Address field of the IPv6 Header equal to its SID address.

  A penultimate SR Segment Endpoint Node is one that, as part of the SID processing, copies the last SID from the SRH into the IPv6 Destination Address and decrements the Segments Left value from one to zero.

  The PSP operation takes place only at a penultimate SR Segment Endpoint Node and does not happen at non-penultimate endpoint nodes. When a SID of PSP-flavor is processed at a non-penultimate SR Segment Endpoint Node, the PSP behavior is not performed since Segments Left would not be zero.

  The SR Segment Endpoint Nodes advertise the SIDs instantiated on them via control plane protocols. A PSP-flavored SID is used by the Source SR Node when it needs to instruct the penultimate SR Segment Endpoint Node listed in the SRH to remove the SRH from the IPv6 header.

- **Ultimate Segment Pop (USP) of the SRH variant**—The SRH processing of the End and End.X behaviors are modified as follows:

  If Segments Left is 0, then:

  1. Update the Next Header field in the preceding header to the Next Header value of the SRH
  2. Decrease the IPv6 header Payload Length by 8*(Hdr Ext Len+1)
  3. Remove the SRH from the IPv6 extension header chain
  4. Proceed to process the next header in the packet

One of the applications of the USP flavor is when a packet with an SRH is destined to an application on hosts with smartNICs implementing SRv6. The USP flavor is used to remove the consumed SRH from the extension header chain before sending the packet to the host.

- **Ultimate Segment Decapsulation (USD) variant**—The Upper-layer header processing of the End and End.X behaviors are modified as follows:
  - **End** behavior: If the Upper-layer Header type is 41 (IPv6), then:
    1. Remove the outer IPv6 Header with all its extension headers
    2. Submit the packet to the egress IPv6 FIB lookup and transmission to the new destination
    3. Else, if the Upper-layer Header type is 4 (IPv4)
    4. Remove the outer IPv6 Header with all its extension headers
    5. Submit the packet to the egress IPv4 FIB lookup and transmission to the new destination
    6. Else, process as per Section 4.1.1 (Upper-Layer Header) of IETF RFC 8986 SRv6 Network Programming
  - **End.X** behavior: If the Upper-layer Header type is 41 (IPv6) or 4 (IPv4), then:
    1. Remove the outer IPv6 Header with all its extension headers
    2. Forward the exposed IP packet to the L3 adjacency J
    3. Else, process as per Section 4.1.1 (Upper-Layer Header) of IETF RFC 8986 SRv6 Network Programming

One of the applications of the USD flavor is the case of TI-LFA in P routers with encapsulation with H.Encaps. The USD flavor allows the last Segment Endpoint Node in the repair path list to decapsulate the IPv6 header added at the TI-LFA Point of Local Repair and forward the inner packet.

## Usage Guidelines and Limitations

### General Guidelines and Limitations

- Cisco IOS XR Release 7.5.2 and later supports the following SRv6 SID behaviors and variants:
  - END with PSP/USD
  - END.X with PSP/USD
  - END.DT4
  - END.DT6
- SRv6 Underlay support includes:
  - IGP redistribution/leaking between levels
  - Prefix Summarization on ABR routers
  - IS-IS TI-LFA
  - Microloop Avoidance

- Flex-algo

### Configuring SRv6

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 usage guidelines and restrictions 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.
- You can configure up to 8 locators to support SRv6 Flexible Algorithm. All locators prefix must share the same SID block (first 40-bits).

#### Enabling SRv6 on the Platform

Before configuring SRv6 on Cisco NCS 540 Series RoutersCisco NCS 540 Series Routers, you must first use the following command in config mode:

- **hw-module profile segment-routing srv6 mode base**

  You must reload the router after enabling this command.

#### Enabling SRv6 with Locator

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

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLoc1
Router(config-srv6-locator)# prefix 2001:db8:0:a2::/64
```

#### (Optional) Configuring SRv6 Anycast Locator

An SRv6 Anycast locator is a type of locator that identifies a set of nodes (END SIDs). SRv6 Anycast Locators and their associated END SIDs may be provisioned at multiple places in a topology.

The set of nodes (Anycast group) is configured to advertise a shared Anycast locator and END SID. Anycast routing enables the steering of traffic toward multiple advertising nodes. Packets addressed to an Anycast address are forwarded to the topologically nearest nodes.

One use case is to advertise Anycast END SIDs at exit points from an SRv6 network. Any of the nodes that advertise the common END SID could be used to forward traffic out of the SRv6 portion of the network to the topologically nearest node.

Unlike a normal locator, IS-IS does not program or advertise END.X SIDs associated with an anycast locator.

**Note** END SIDs allocated from Anycast locators will not be used in constructing TI-LFA backup paths or Microloop Avoidance primary paths. TI-LFA backup and Microloop Avoidance paths for an Anycast locator prefix may terminate on any node advertising that locator, which may be different from the node terminating the original primary path.

**Note** SRv6 anycast locators may have non-zero algorithm (Flexible Algorithm) values.

The following example shows how to globally enable SRv6 and configure Anycast locator.

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLoc1 anycast
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:

- **segment-routing srv6 encapsulation source-address** *ipv6-addr*—Source Address of outer encapsulating IPv6 header. The default source address for encapsulation is one of the loopback addresses.
- **segment-routing srv6 encapsulation hop-limit** {*count* | **propagate**}—The hop limit of outer-encapsulating IPv6 header. The range for *count* is from 1 to 254; the default value for hop-limit is 254. Use **propagate** to set the hop-limit value by propagation (from incoming packet/frame).

```
Router(config)# segment-routing srv6
Router(config-srv6)# encapsulation source-address 1::1
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.

```
Router(config)# segment-routing srv6
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.

```
Router# show segment-routing srv6 manager
Parameters:
  Parameters:
  SRv6 Enabled: Yes
  SRv6 Operational Mode:
    Base:
      SID Base Block: 2001:db8::/40
  Encapsulation:
    Source Address:
      Configured: 1::1
      Default: 5::5
```

```
    Hop-Limit: Default
    Traffic-class: 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
        History: (0 cleared, 0 warnings, 0 full)
    Block 2001:db8:0:a2::/64:
        Number of SIDs free: 65470
        Max SIDs: 65470
        Thresholds: Green 3274, Warning 1965
        Status: Resource Available
            History: (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: Yes
  Microloop-Avoidance: Yes
  Endpoint behaviors:
    End (PSP)
    End.X (PSP)
    End.DX6
    End.DX4
    End.DT6
    End.DT4
    End.DX2
    uN (PSP/USD)
    uA (PSP/USD)
    uDT6
    uDT4
    uDX2
    uB6 (Insert.Red)
  Headend behaviors:
    T
    H.Insert.Red
    H.Encaps.Red
  Security rules:
    SEC-1
    SEC-2
    SEC-3
  Counters:
    CNT-1
    CNT-3
  Signaled parameters:
    Max-SL          : 3
    Max-End-Pop-SRH : 3
    Max-H-Insert    : 3 sids
    Max-H-Encap     : 3 sids
    Max-End-D       : 4
  Configurable parameters (under srv6):
    Encapsulation:
      Source Address: Yes
      Hop-Limit     : value=Yes, propagate=No
      Traffic-class : value=Yes, propagate=Yes
  Max SIDs: 64000
  SID Holdtime: 3 mins
```

### Verifying SRv6 Locator

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

```
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 Local SIDs

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

```
Router# show segment-routing srv6 locator myLoc1 sid

SID                         Function     Context                           Owner
     State  RW
--------------------------  -----------  ------------------------------    ------------------
  -----  --
2001:db8:0:a2:1::           End (PSP)    'default':1                       sidmgr
     InUse  Y
2001:db8:0:a2:40::          End.DT4      'VRF1'                            bgp-100
     InUse  Y
2001:db8:0:a2:41::          End.X (PSP)  [Hu0/1/0/1, Link-Local]           isis-srv6
     InUse  Y
```

The following example shows how to display detail information regarding an allocated SRv6 local SID.

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

SID                         Function     Context                           Owner
     State  RW
--------------------------  -----------  ------------------------------    ------------------
  -----  --
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:

| Command | Description |
|---|---|
| **show segment-routing srv6 manager** | Displays the summary information from SRv6 manager, including platform capabilities. |
| **show segment-routing srv6 locator** *locator-name* [**detail**] | Displays the SRv6 locator information on the router. |

| Command | Description |
|---|---|
| **show segment-routing srv6 locator** *locator-name* **sid** [[*sid-ipv6-address* [**detail**] | Displays the information regarding SRv6 local SID(s) allocated from a given locator. |
| **show segment-routing srv6 sid** [*sid-ipv6-address* \| **all** \| **stale**] [**detail**] | Displays SID information across locators. By default, only "active" (i.e. non-stale) SIDs are displayed. |
| **show route ipv6 local-srv6** | Displays all SRv6 local-SID prefixes in IPv6 RIB. |

# SRv6 Micro-Segment (uSID)

***Table 3: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SRv6 Micro-Segment (uSID) | Release 7.3.1 | This feature is an extension of the SRv6 architecture. It leverages the existing SRv6 Network Programming architecture to encode up to six SRv6 Micro-SID (uSID) instructions within a single 128-bit SID address. Such a SID address is called a uSID Carrier.<br>In addition, this feature leverages the existing SRv6 data plane and control plane with no changes. It also provides low MTU overhead; for example, 6 uSIDs per uSID carrier results in 18 source-routing waypoints in only 40 bytes of overhead (in SRH). |

The SRv6 micro-segment (uSID) is an extension of the SRv6 architecture. It leverages the SRv6 Network Programming architecture to encode several SRv6 Micro-SID (uSID) instructions within a single 128-bit SID address. Such a SID address is called a uSID Carrier.

SRv6 uSID is documented in the IETF drafts Network Programming extension: SRv6 uSID instruction and Compressed SRv6 Segment List Encoding in SRH.

Throughout this chapter, we will refer to SRv6 micro-segment as "uSID".

The SRv6 uSID provides the following benefits:

- Leverages the SRv6 Network Programming with no change. SRv6 uSID is a new pseudo code in the existing SRv6 network programming framework.
- Leverages the SRv6 data plane (SRH) with no change. Any SID in the destination address or SRH can be an SRv6 uSID carrier.
- Leverages the SRv6 control plane with no change.
- Ultra-Scale—Scalable number of globally unique nodes in the domain, for example:
  - 16-bit uSID ID size: 65k uSIDs per domain block
  - 32-bit uSID ID size: 4.3M uSIDs per domain block
- Lowest MTU overhead

- 6 uSIDs per uSID carrier
- For example, 18 source-routing waypoints in only 40 bytes of overhead

- Hardware-friendliness:
  - Leverages mature hardware capabilities (inline IP Destination Address edit, IP Destination Address longest match).
  - Avoids any extra lookup in indexed mapping tables.
  - A micro-program with 6 or fewer uSIDs requires only legacy IP-in-IP encapsulation behavior.
- Scalable Control Plane:
  - Summarization at area/domain boundary provides massive scaling advantage.
  - No routing extension is required, a simple prefix advertisement suffices.
- Seamless Deployment:
  - A uSID may be used as a SID (the carrier holds a single uSID).
  - The inner structure of an SR Policy can stay opaque to the source. A carrier with uSIDs is just seen as a SID by the policy headend Security.
  - Leverages SRv6's native SR domain security.

# SRv6 uSID Terminology

The SRv6 Network Programming is extended with the following terms:

- uSID—An identifier that specifies a micro-segment.

  A uSID has an associated behavior that is the SRv6 function (for example, a node SID or Adjacency SID) associated with the given ID. The node at which an uSID is instantiated is called the “Parent” node.

- uSID Carrier—A 128-bit IPv6 address (carried in either in the packet destination address or in the SRH) in the following format:

  ```
  <uSID-Block><Active-uSID><Next-uSID>...<Last-uSID><End-of-Carrier>...<End-of-Carrier>
  ```

  where:

  - uSID Block—An IPv6 prefix that defines a block of SRv6 uSIDs.
  - Active uSID—The first uSID that follows the uSID block.
  - Next uSID—The next uSID after the Active uSID.
  - Last uSID—The last uSID in the carrier before the End-of-Carrier uSID.
  - End-of-Carrier —A globally reserved uSID that marks the end of a uSID carrier. The End-of-Carrier ID is **0000**. All empty uSID carrier positions must be filled with the End-of-Carrier ID; therefore, a uSID carrier can have more than one End-of-Carrier.

  The following is an example of an SRH with 3 Micro-SID carriers for a total of up to 18 micro-instructions:

| |
|---|
| Micro-SID Carrier1: {uInstruction1, uInstruction2… uInstruction6} |
| Micro-SID Carrier2: {uInstruction7, uInstruction8… uInstruction12} |
| Micro-SID Carrier3: {uInstruction13, uInstruction14… uInstruction18} |

# SRv6 uSID Carrier Format

The uSID carrier format specifies the type of uSID carrier supported in an SRv6 network. The format specification includes Block size and ID size.

- **uSID Block**

  The uSID block is an IPv6 prefix that defines a block of SRv6 uSIDs. This can be an IPv6 prefix allocated to the provider (for example, /22, /24, and so on.), or it can be any well-known IPv6 address block generally available for private use, such as the ULA space FC/8, as defined in IETF draft RFC4193.

  An SRv6 network may support more than a single uSID block.

  The length of block [prefix] is defined in bits. From a hardware-friendliness perspective, it is expected to use sizes on byte boundaries (16, 24, 32, and so on).

- **uSID ID**

  The length of uSID ID is defined in bits. From a hardware-friendliness perspective, it is expected to use sizes on byte boundaries (8, 16, 24, 32, and so on).

The uSID carrier format is specified using the notation "F*bbuu*" , where “*bb*” is size of block and “*uu*” is size of ID. For example, "F3216" is a format with a 32-bit uSID block and 16-bit uSID IDs.

**Note** F3216 is the default format, and the only format that is supported in IOS XR 7.3.1 release.

# SRv6 uSID Allocation Within a uSID Block

The architecture for uSID specifies both globally scoped and locally scoped uSIDs, where a globally scoped uSID is the type of uSID that provides reachability to the node.

On the other hand, a locally scoped uSID is associated to a local behavior, and therefore *must* be preceded by a globally scoped uSID of the parent node when relying on routing to forward the packet.

The Global ID block (GIB) is the set of IDs available for globally scoped uSID allocation. The Local ID block (LIB) is the set of IDs available for locally scoped uSID allocation.

A globally scoped uSID is a uSID from the GIB. A globally scoped uSID typically identifies a shortest path to a node in the SR domain. An IP route (for example, /48) is advertised by the parent node to each of its globally scoped uSIDs, under the associated uSID block. The parent node executes a variant of the END behavior.

The “Nodal” uSID (uN) is an example of a globally scoped behavior defined in uSID architecture.

A node can have multiple globally scoped uSIDs under the same uSID blocks (for example, one per IGP flex-algorithm). Multiple nodes may share the same globally scoped uSID (Anycast).

A locally scoped uSID is a uSID from the LIB. A locally scoped uSID identifies a local micro-instruction on the parent node; for example, it may identify a cross-connect to a direct neighbor over a specific interface or a VPN context. Locally scoped uSIDs are not routeable.

For example, if N1 and N2 are two different physical nodes of the uSID domain and *L* is a locally scoped uSID value, then N1 and N2 may bind two different behaviors to *L*.

The uSIDs are allocated in one of following ways: auto, dynamic, or explicit.

- The request to allocate locally scoped uSIDs comes from SRv6 clients (such as IS-IS or BGP). The request can be to allocate any available ID (dynamic allocation) or to allocate a specific ID (explicit allocation).

# SRv6 Endpoint Behaviors Associated with uSID

The SRv6 Network Programming is extended with new types of SRv6 SID endpoint behaviors:

- uN—A short notation for the NEXT-CSID (Compressed SID) End behavior with a pseudocode of shift-and-lookup, and PSP/USD flavors
- uA—A short notation for the NEXT-CSID End.X behavior with a pseudocode of shift-and-xconnect, and PSP/USD flavors
- uDT—A short notation for the NEXT-CSID End.DT behavior with the same pseudocode as End.DT4/End.DT6/End.DT46/End.DT2U/End.DT2M
- uDX—A short notation for the NEXT-CSID End.DX behavior with the same pseudocode as End.DX4/End.DX6/End.DX2

# SRv6 uSID in Action - Example

This example highlights an integrated VPN and Traffic Engineering use-case leveraging SRv6 uSID.

VPNv4 site A connected to Node 1 sends packets to VPNv4 site B connected to Node 2 alongside a traffic engineered path via Node 8 and Node 7 using a single 128-bit SRv6 SID.

Node 1 is the ingress PE; Node 2 is the egress PE.

Nodes 3, 4, 5, and 6 are classic IPv6 nodes. Traffic received on these nodes use classic IP forwarding without changing the outer DA.

Nodes 1, 8, 7 and 2 are SRv6 capable configured with:

- 32-bit SRv6 block = fcbb:bb01
- 16-bit SRv6 ID

  For example:

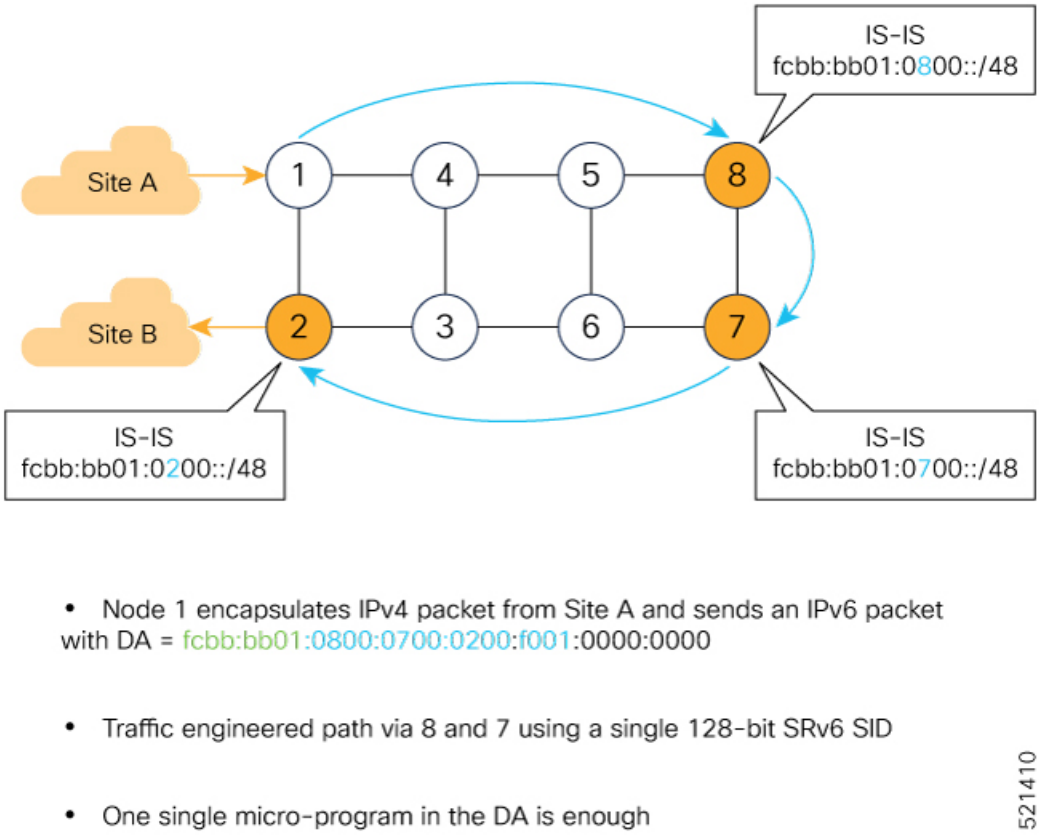
  - Node 7 uN = fcbb:bb01:0700::/48
  - Node 8 uN = fcbb:bb01:0800::/48

The following IGP routes are advertised:

- Node 8 advertises the IGP route fcbb:bb01:**0800**::/48

- Node 7 advertises the IGP route fcbb:bb01:**0700**::/48
- Node 2 advertises the IGP route fcbb:bb01:**0200**::/48

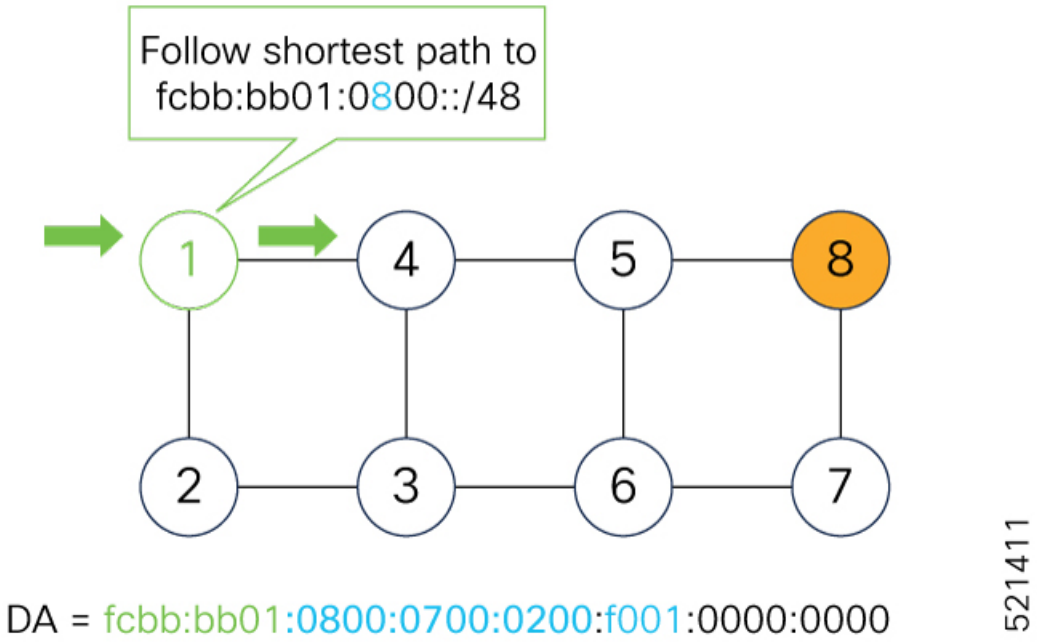
***Figure 2: Integrated VPN and Traffic Engineering SRv6 uSID Use-case***




Node 1 encapsulates an IPv4 packet from VPN Site A and sends an IPv6 packet with destination address fcbb:bb01:**0800**:**0700**:**0200**:f001:0000:0000. This is a uSID carrier, with a list of micro-instructions (uSIDs) (0800, 0700, 0200, f001, and 0000 – indicating the end of the instruction).

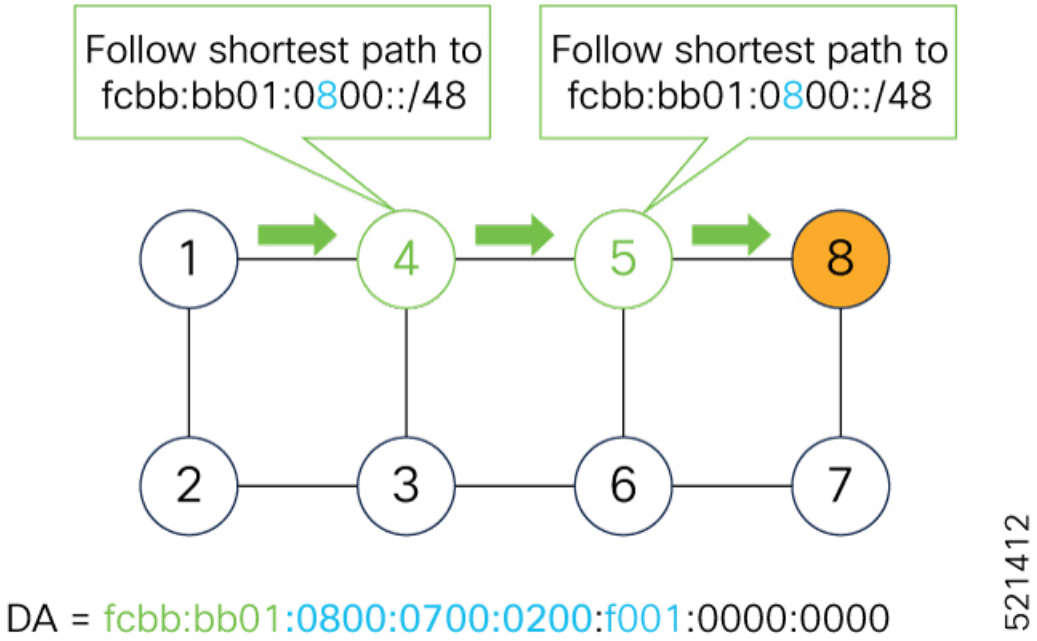
uSIDs (uNs) 0800, 0700, 0200 are used to realize the traffic engineering path to Node 2 with way points at Nodes 8 and 7. uSID f001 is the BGP-signalled instruction (uDT4) advertized by Node 2 for the VPNv4 service

*Figure 3: Node 1: End.B6.Encaps Behavior*




Nodes 4 and 5 simply forward the packet along the shortest path to Node 8, providing seamless deployment through classic IPv6 nodes.

*Figure 4: Node 4 and Node 5: Classic IPv6 Nodes*

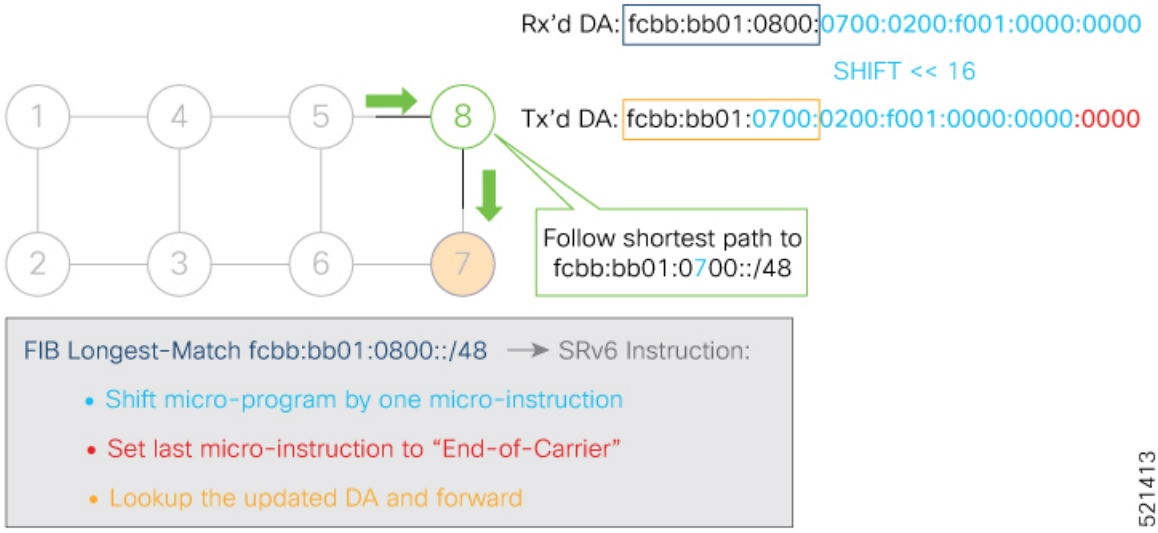



When Node 8 receives the packet, it performs SRv6 uN behavior (shift-and-lookup with PSP/USD). It removes its outer DA (0800) and advances the micro program to the next micro instruction by doing the following:

1. Pops its own uSID (0800)

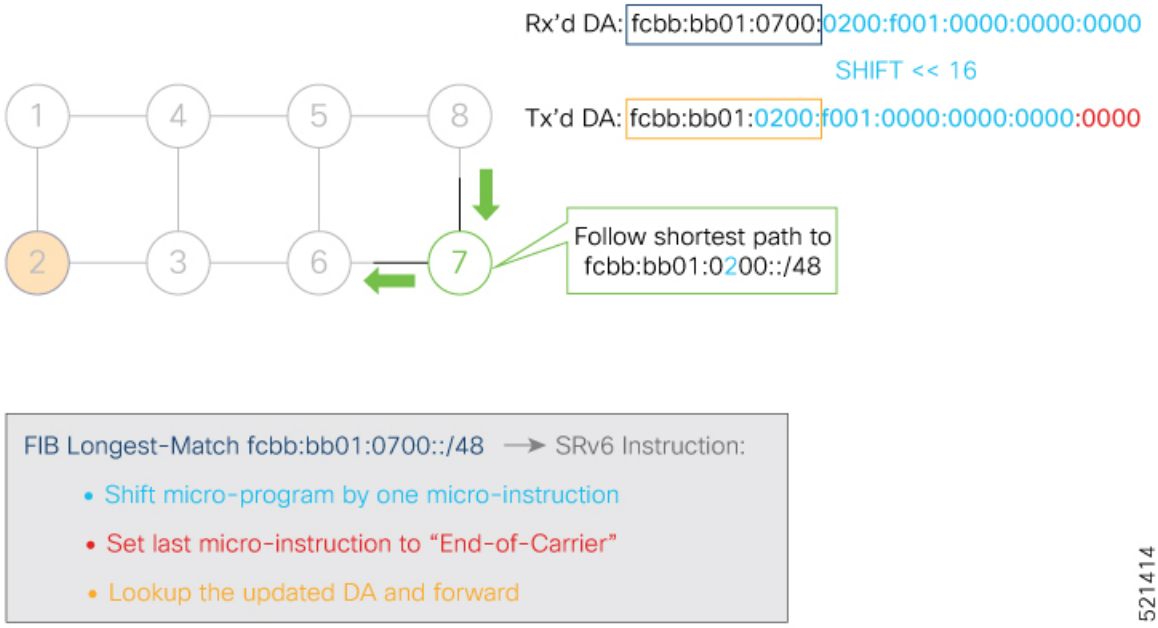
2. **Shifts** the remaining DA by 16-bits to the left
3. Fills the remaining bits with 0000 (End-of-Carrier)
4. Performs a **lookup** for the shortest path to the next DA (fcbb:bb01:**0700::/48**)
5. Forwards it using the new DA fcbb:bb01:**0700**:0200:f001:0000:0000:0000

***Figure 5: Node 8: SRv6 uN Behavior (Shift and Forward)***



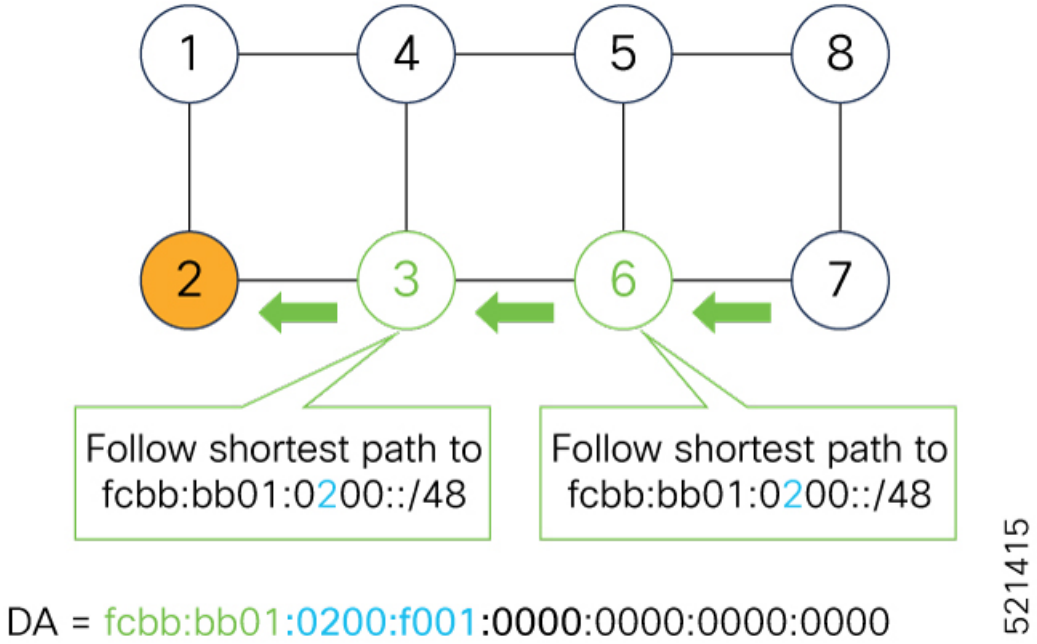

When Node 7 receives the packet, it performs the same SRv6 uN behavior (shift-and-lookup with PSP/USD), forwarding it using the new DA fcbb:bb01:**0200**:f001:0000:0000:0000:0000

***Figure 6: Node 7: SRv6 uN Behavior (Shift and Forward)***



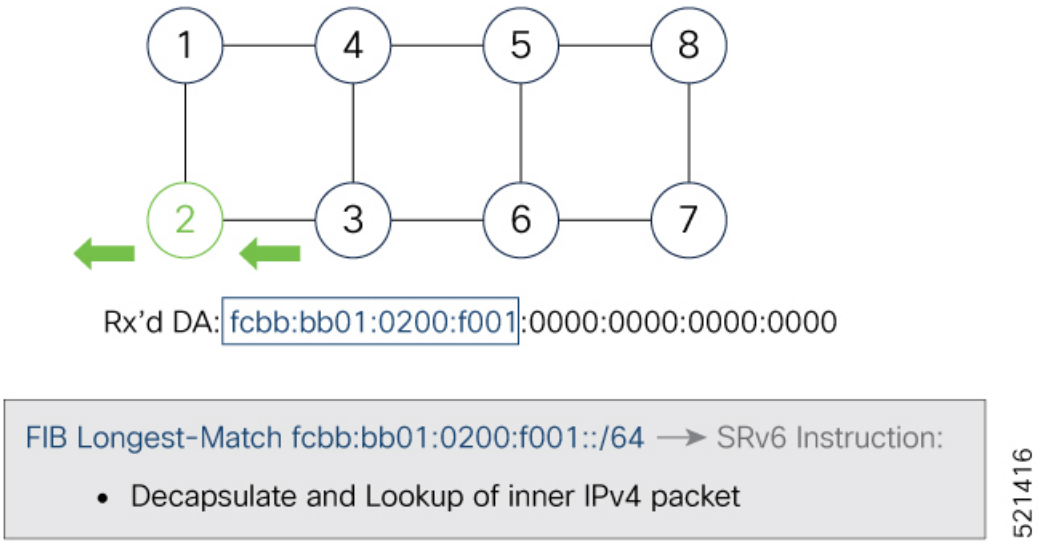
Nodes 6 and 3 simply forward the packet along the shortest path to Node 2, providing seamless deployment through classic IPv6 nodes.

***Figure 7: Node 6 and Node 3: Classic IPv6 Nodes***




When Node 2 receives the packet, it performs an SRv6 uDT4 behavior (End.DT4—Endpoint with decapsulation and IPv4 table lookup) to VPNv4 Site B.

***Figure 8: Node 2: SRv6 uDT4 Behavior***




To recap, this example showed an integrated VPN and Traffic Engineering use-case, where VPNv4 site A connected to Node 1 sent packets to VPNv4 site B connected to Node 2 alongside a traffic engineered path via Node 8 and Node 7 using a single 128-bit SRv6 SID:

- @1: inner packet P encapsulated with outer DA fcbb:bb01:**0800**:**0700**:**0200**:f001:0000:0000

- @4 & @5: classic IP forwarding, outer DA unchanged
- @8: SRv6 uN behavior: shift and lookup, outer DA becomes fcbb:bb01:**0700**:**0200**:f001:0000:0000:0000
- @7: SRv6 uN behavior: shift and lookup, outer DA becomes fcbb:bb01:**0200**:f001:0000:0000:0000
- @6 & @3: classic IP forwarding, outer DA unchanged
- @2: SRv6 End.DT4: Decapsulate and IPv4 table lookup

# Usage Guidelines and Limitations

### General Guidelines and Limitations

- Cisco IOS XR supports uSIDs with 32-bit uSID block and 16-bit uSID IDs (3216).

  A single UCF format must be used for uSID locators in a SRv6 uSID domain.

- Cisco IOS XR supports up to 8 uSID locator prefixes.

  Multiple locator prefixes are used when configuring Anycast locators or SRv6 Flexible Algorithm instances, for example.

- Cisco IOS XR supports uSID locator prefixes from different uSID blocks.

  Up to 64 uSID blocks can be used across all uSID locators in the network.

- Cisco IOS XR Release 7.3.1 and later supports the following SRv6 uSID behaviors and variants:
  - uN with PSP/USD
  - uA with PSP/USD
  - uDT4
  - uDT6

- SRv6 Underlay support includes:
  - IGP redistribution/leaking between levels
  - Prefix Summarization on ABR routers
  - IS-IS TI-LFA
  - Microloop Avoidance
  - Flex-algo

- SRv6 over GRE interface is not supported
- SRv6 is not supported on the following NCS 540 router variants, on ports 16, 17, 18, and 19:
  - N540X-4Z14G2Q-A
  - N540X-4Z14G2Q-D

- In situations where SRv6 has been previously configured, attempting to configure an IPv6 ACL subsequently results in failure for the following NCS 540 router variants. It's important to note that these two features cannot co-exist; only one can be active at any given time.
  - N540X-6Z18G-SYS-A
  - N540X-6Z18G-SYS-D
  - N540X-8Z16G-SYS-A
  - N540X-8Z16G-SYS-D
  - N540X-4Z14G2Q-A
  - N540X-4Z14G2Q-D
  - N540-6Z18G-SYS-A
  - N540-6Z18G-SYS-D

#### uSID Allocation Recommendation

We recommend that the uSID block allocation is made from the IPv6 Unique Local Address (ULA) range.

**Note** Allocation from the public Global Unicast Addresses (GUA) range is also supported.

- Use ULA /24 base from FC00::/8 space
  - FC*BB*:*BB*/24, with *B* indicating a nibble value picked by operator
- 256 uSID blocks possible from this allocation
  - In this release, 64 uSID blocks are supported
  - FC*BB*:*BBVV*/32, with *VV* two variable nibbles. The supported values for *VV* in Cisco IOS XR Release 7.3.1 are 0x00 to 0x3F.

For example:

- ULA /24 base = FC00:01/24
- uSID block space = 64 uSID blocks (from FC00:01**00**/32 to FC00:01**3F**/32)

#### Platform-Specific Guidelines and Limitations

# Configuring SRv6

Enabling SRv6 involves the following high-level configuration steps:

- Enable SRv6 on the platform
- Configure SRv6 locator(s)

- Enable SRv6 under IS-IS
- Enable SRv6 Services under BGP

### Enable SRv6 on the Platform

Before configuring SRv6 on the Cisco NCS 540 Series Routers router, you must first use the following command:

- **hw-module profile segment-routing srv6 mode micro-segment format f3216**

**Note** You must reload the router after enabling this command.

### (Optional) Configure Network Role

By default, after enabling SRv6 on the platform, the node can serve as both edge (services) and core roles.

Optionally, you can customize the node role as "core-only" using the following command:

- **hw-module profile network-role core-only**

**Note** You must reload the router after enabling this command.

Given that there is different budget for underlay SID encap based on the node role in the network (P-only vs Edge), an operator can use this configuration to provide a hint to the platform and control plane to use a larger SID encap budget when operating as a P-only node.

### (Optional) Configure Merge Overlay/Underlay SID Mode

One of the main benefits of SRv6 uSID is compression (or packing) of multiple uSIDs into a uSID carrier. This is possible when they share the same uSID block and when there is enough space in the carrier.

The underlay SIDs are always programmed in compressed form, if possible. The overlay SID is programmed separately.

The **segment-routing srv6 micro-segment merge-overlay-underlay-sids** command is used to enable the platform to merge overlay/underlay SIDs.

**Note** From Cisco IOS XR Release 7.7.1 and later, the compression/merging of uSID lists is automatically done in the dataplane during imposition. This configuration option is ignored by the dataplane.

When there is a need to send overlay traffic, the data path implementation attempts to merge the underlay SIDs and overlay SIDs into a single carrier, if possible. With H.Encaps.Red encapsulation, this yields a packet with no SRH.

**Note** If the overlay and underlay use different uSID blocks, this merge is not possible.

By default, the Cisco NCS platform does not automatically merge the overlay/underlay SIDs.

To enable the platform to merge overlay/underlay SIDs, use the following command:

- **segment-routing srv6 micro-segment merge-overlay-underlay-sids**

**Caution** This command should only be enabled when a single block is required.

After you enable this command, this CLI will modify the behavior for all new overlay routes being programmed afterwards.

If you enable this command after SRv6 overlay routes are already programmed, we recommend that you clear the SRv6 overlay routes (using the **clear route** [**vrf** *WORD*] command) in order to trigger the re-programming in the "merge" mode.

If you do not to clear the overlay routes, those routes would continue to be programmed in the "non-merge" mode.

**Note**

- When you downgrade from Cisco IOS XR Software Release 7.6.2 and above to Cisco IOS XR Software Release 7.3.2, you must configure **hw-module profile segment-routing srv6 mode micro-segment format f3216 encapsulation traffic-class propagate** because the following commands fail to load:
  - hw-module profile segment-routing srv6 mode micro-segment format f3216 encapsulation l2-traffic traffic-class propagate
  - hw-module profile segment-routing srv6 mode micro-segment format f3216 encapsulation l3-traffic traffic-class propagate
- You must reload the line card after enabling this command.

### Configure SRv6 Locator Name, Prefix, and uSID-Related Parameters

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

- **segment-routing srv6 locators locator** *locator*—Globally enable SRv6 and configure the locator.
- **segment-routing srv6 locators locator** *locator* **prefix** *ipv6_prefix*/*length*—Configure the locator prefix value.
- **segment-routing srv6 locators locator** *locator* **micro-segment behavior unode psp-usd**—Specifies the locator as a micro-segment (uSID) locator as well as specifies that IGP underlay uSID (uN/uA) variant is PSP-USD for this locator.

### (Optional) Configure Algorithm Associated with Locator

- **segment-routing srv6 locators locator** *locator* **algorithm** *algo*—(Optional) Configure Algorithm associated with the locator. Valid values for *algo* are from 128 to 255.

  For additional information about SRv6 Flexible Algorithm, see Configuring SRv6 Flexible Algorithm under IS-IS, on page 31.

For detailed information about Flexible Algorithm, see Enabling Segment Routing Flexible Algorithm, on page 409.

### (Optional) Configure Anycast Locator

An SRv6 Anycast locator is a type of locator that identifies a set of nodes (uN SIDs). SRv6 Anycast Locators and their associated uN SIDs may be provisioned at multiple places in a topology.

The set of nodes (Anycast group) is configured to advertise a shared Anycast locator and uN SID. Anycast routing enables the steering of traffic toward multiple advertising nodes. Packets addressed to an Anycast address are forwarded to the topologically nearest nodes.

One use case is to advertise Anycast uN SIDs at exit points from an SRv6 network. Any of the nodes that advertise the common uN SID could be used to forward traffic out of the SRv6 portion of the network to the topologically nearest node.

The following behaviors apply to Anycast Locator:

- Unlike a normal locator, IS-IS does not program or advertise uA SIDs associated with an Anycast locator.
- uN SIDs allocated from Anycast locators will not be used in constructing TI-LFA backup paths or Microloop Avoidance primary paths. TI-LFA backup and Microloop Avoidance paths for an Anycast locator prefix may terminate on any node advertising that locator, which may be different from the node terminating the original primary path.
- SRv6 anycast locators may have non-zero algorithm (Flexible Algorithm) values.

Use the following commands to configure the Anycast locator and advertise Anycast prefixes associated with an interface.

- **segment-routing srv6 locators locator** *locator* **anycast**—Configure the Anycast locator
- **router isis** *instance-id* **interface Loopback** *instance* **prefix-attributes anycast level** *level*—Advertise the Anycast prefixes associated with an interface.

**Example 1:**

The following example shows how to globally enable SRv6 and configure a locator.

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLoc1
Router(config-srv6-locator)# micro-segment behavior unode psp-usd
Router(config-srv6-locator)# prefix 2001:0:8::/48
```

**Example 2:**

The following example shows how to configure Flexible Algorithm associated with locator.

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLocAlgo128
Router(config-srv6-locator)# algorithm 128
Router(config-srv6-locator)# micro-segment behavior unode psp-usd
Router(config-srv6-locator)# prefix 2001:0:88::/48
```

**Example 3:**

The following example shows how to configure Anycast locator.

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLocAnycast
Router(config-srv6-locator)# anycast
Router(config-srv6-locator)# micro-segment behavior unode psp-usd
Router(config-srv6-locator)# prefix 2001:0:100::/48
```

The following example shows how to advertise the Anycast prefixes associated with an interface.

```
Router(config)# router isis core
Router(config-isis)# interface Loopback100
Router(config-isis-if)# prefix-attributes anycast level 1
```

### (Optional) Customize SRv6 Encapsulation Parameters

This section describes the configurable SRv6 encapsulation parameters. These optional parameters include:

- **segment-routing srv6 encapsulation source-address** *ipv6-addr*—Source Address of outer encapsulating IPv6 header. The default source address for encapsulation is one of the loopback addresses.
- **segment-routing srv6 encapsulation hop-limit** {*count* | **propagate**}—The hop limit of outer-encapsulating IPv6 header. The range for *count* is from 1 to 254; the default value for hop-limit is 254. Use **propagate** to set the hop-limit value by propagation (from incoming packet/frame).
- **segment-routing srv6 encapsulation evpn next-header** *protocol-number*—The protocol number to use in the Next-header field of the IPv6 or SRH header. The range for *protocol-number* is from 59 to 252.

### (Optional) Customize SRv6 Logging for Locator Status Changes

- **segment-routing srv6 logging locator status**—Enable the logging of locator status.

### (Optional) Customize SRv6 SID Parameters

- **segment-routing srv6 sid holdtime** *minutes*—The holdtime for a stale or freed SID. The range of *minutes* is from 0 (disabled) to 60 minutes.

**Example 4:**

The following example shows how to configure optional SRv6 parameters:

```
RP/0/RSP0/CPU0:Node1(config)# segment-routing srv6 encapsulation
RP/0/RSP0/CPU0:Node1(config-srv6-encap)# source-address 1::1
RP/0/RSP0/CPU0:Node1(config-srv6-encap)# hop-limit 60
RP/0/RSP0/CPU0:Node1(config-srv6-encap)# evpn next-header 65
RP/0/RSP0/CPU0:Node1(config-srv6-encap)# exit
RP/0/RSP0/CPU0:Node1(config-srv6)# logging locator status
RP/0/RSP0/CPU0:Node1(config-srv6)# sid holdtime 10
RP/0/RSP0/CPU0:Node1(config-srv6)# micro-segment merge-overlay-underlay-sids

This config applies to only new SRv6 micro-segment overlay routes and does not update already
 programmed routes.
Please flap any existing SRv6 micro-segment overlay routes after making this configuration
 change.

RP/0/RSP0/CPU0:Node1(config-srv6)#
```

### 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.

```
Router# show segment-routing srv6 manager
Parameters:
  SRv6 Enabled: Yes
  SRv6 Operational Mode:
    Micro-segment:
      SID Base Block: 2001::/24
  Encapsulation:
    Source Address:
      Configured: ::
      Default: ::
    Hop-Limit: Default
    Traffic-class: Default
Summary:
  Number of Locators: 3 (3 operational)
  Number of SIDs: 3 (0 stale)
  Max SIDs: 64000
  OOR:
    Thresholds: Green 3200, Warning 1920
    Status: Resource Available
        History: (0 cleared, 0 warnings, 0 full)
    Block 2001::/32:
        Number of SIDs free: 7680
        Max SIDs: 7680
        Thresholds: Green 384, Warning 231
        Status: Resource Available
            History: (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: Yes
  Microloop-Avoidance: Yes
  Endpoint behaviors:
    End (PSP)
    End.X (PSP)
    End.DX6
    End.DX4
    End.DT6
    End.DT4
    uN (PSP/USD)
    uA (PSP/USD)
    uDT6
    uDT4
    uDX2
    uB6 (Insert.Red)
  Headend behaviors:
    T
    H.Insert.Red
    H.Encaps.Red
  Security rules:
    SEC-1
    SEC-2
    SEC-3
  Counters:
    CNT-1
    CNT-3
  Signaled parameters:
    Max-SL          : 3
    Max-End-Pop-SRH : 3
    Max-H-Insert    : 3 sids
    Max-H-Encap     : 3 sids
```

```
    Max-End-D       : 4
  Configurable parameters (under srv6):
    Encapsulation:
      Source Address: Yes
      Hop-Limit     : value=Yes, propagate=No
      Traffic-class : value=Yes, propagate=Yes
  Max SIDs: 64000
  SID Holdtime: 3 mins
```

### Verifying SRv6 Locator

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

```
Router# show segment-routing srv6 locator myLoc1 detail

Name                  ID       Algo  Prefix                    Status   Flags
--------------------  -------  ----  ------------------------  -------  --------
myLoc1                3        0     2001:0:8::/48             Up       U
  (U): Micro-segment (behavior: uN (PSP/USD))
  Interface:
    Name: srv6-myLoc1
    IFH : 0x02000120
    IPv6 address: 2001:0:8::/48
  Number of SIDs: 1
  Created: Dec 10 21:26:54.407 (02:52:26 ago)
```

### Verifying SRv6 SIDs

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

```
Router# show segment-routing srv6 locator myLoc1 sid
SID                         Behavior          Context                           Owner
          State  RW
--------------------------  ----------------  ------------------------------
------------------  -----  --
2001:0:8::                  uN (PSP/USD)      'default':1                       sidmgr
          InUse  Y
```

The following example shows how to display detail information regarding an allocated SRv6 local SID.

```
Router# show segment-routing srv6 locator myLoc1 sid 2001:0:8:: detail
SID                         Behavior          Context                           Owner
          State  RW
--------------------------  ----------------  ------------------------------
------------------  -----  --
2001:0:8::                  uN (PSP/USD)      'default':8                       sidmgr
          InUse  Y
  SID Function: 0x8
  SID context: { table-id=0xe0800000 ('default':IPv6/Unicast), opaque-id=8 }
  Locator: 'myLoc1'
  Allocation type: Dynamic
  Created: Dec 10 22:10:51.596 (02:10:05 ago)
```

Similarly, you can display SID information across locators by using the **show segment-routing srv6 sid** command.

# Configuring SRv6 under 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 IS-IS 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.

### Usage Guidelines and Restrictions

The following usage guidelines and restrictions apply for SRv6 IS-IS:

- An IS-IS address-family can support either SR-MPLS or SRv6, but both at the same time is not supported.

### Configuring SRv6 under IS-IS

To configure SRv6 IS-IS, use the **router isis** command. Enable SRv6 under the IS-IS IPv6 address-family and assign SRv6 locator(s) to it. Use the **level** {**1** | **2**} keywords to advertise the locator only in the specified IS-IS level.

**Note** If no level is specified, local locators will be advertised into all configured ISIS levels. Ensure that locators are included in the redistribution or propagation policy to prevent potential loops when redistributing between multiple instances or propagating between Level 2 and Level 1.

The following example shows how to configure SRv6 under IS-IS.

```
Router(config)# router isis core
Router(config-isis)# address-family ipv6 unicast
Router(config-isis-af)# segment-routing srv6
Router(config-isis-srv6)# locator myLoc1 level 1
Router(config-isis-srv6-loc)# exit
```

For more information about configuring IS-IS, refer to the "*Implementing IS-IS*" chapter in the *Routing Configuration Guide for Cisco NCS 540*.

# Configuring SRv6 Flexible Algorithm under IS-IS

This feature introduces support for implementing Flexible Algorithm using IS-IS SRv6.

SRv6 Flexible Algorithm allows operators to customize IGP shortest path computation according to their own needs. An operator can assign custom SR prefix-SIDs to realize forwarding beyond link-cost-based SPF. As a result, Flexible Algorithm provides a traffic-engineered path automatically computed by the IGP to any destination reachable by the IGP.

For detailed information about Flexible Algorithm, see Enabling Segment Routing Flexible Algorithm, on page 409.

### Usage Guidelines and Restrictions

Observe the following usage guidelines and restrictions:

- You can configure up to 8 locators to support SRv6 Flexible Algorithm.
- The Flexible Algorithm locator prefix follows the same usage guidelines and restrictions of algo-0 locator prefixes. See Usage Guidelines and Limitations, on page 23.
- The Locator Algorithm value range is 128 to 255.

### Configuring SRv6 Flexible Algorithm under IS-IS

The following sections show you the steps to enable SRv6 Flexible Algorithm. The example highlights a delay-based Flexible Algorithm instance.

1. Configure SRv6 locators
2. Assign SRv6 locators under IS-IS
3. Configure Flexible Algorithm definition and associated metric (for example, delay)
4. Configure the delay probe under the interface. For more information on SR performance measurement, see Configure performance measurement.

The following section shows how to configure two SRv6 locators: one associated with Algo 0, and the other associated with Algo 128.

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLocBestEffort // best-effort locator
Router(config-srv6-locator)# micro-segment behavior unode psp-usd
Router(config-srv6-locator)# prefix 2001:0:1::/48
Router(config-srv6-locator)# exit

Router(config-srv6-locators)# locator myLocLowLat // low-latency (flex algo 128) locator
Router(config-srv6-locator)# micro-segment behavior unode psp-usd
Router(config-srv6-locator)# prefix 2001:0:2::/48
Router(config-srv6-locator)# algorithm 128
Router(config-srv6-locator)# exit
Router(config-srv6)# exit
```

The following section shows how to assign multiple SRv6 locators under IS-IS.

```
Router(config)# router isis core
Router(config-isis)# address-family ipv6 unicast
Router(config-isis-af)# segment-routing srv6
Router(config-isis-srv6)# locator myLocBestEffort
Router(config-isis-srv6-loc)# exit
```

```
Router(config-isis-srv6)# locator myLocLowLat
Router(config-isis-srv6-loc)# exit
```

The following section shows how to configure the Flexible Algorithm definition.

```
Router(config)# router isis core
Router(config-isis)# flex-algo 128
Router(config-isis-flex-algo)# metric-type delay
Router(config-isis-flex-algo)# exit
Router(config-isis)# interface GigabitEthernet0/0/0/0
Router(config-isis-if)# address-family ipv6 unicast
```

The following section shows how to configure the delay probe under the interface.

```
Router(config)# performance-measurement
Router(config-perf-meas)# interface GigabitEthernet0/0/0/0
Router(config-pm-intf)# delay-measurement
Router(config-pm-intf-dm)# commit
```

## Verification

```
Router# show segment-routing srv6 locator

Name                  ID       Algo  Prefix                    Status   Flags
--------------------  -------  ----  ------------------------  -------  --------
myLoc1                3        0     2001:0:8::/48             Up       U
myLocBestEffort       5        0     2001:0:1::/48             Up       U
myLocLowLat           4        128   2001:0:2::/48             Up       U


Router# show isis flex-algo 128

IS-IS core Flex-Algo Database

Flex-Algo 128:

Level-2:
        Definition Priority: 128
        Definition Source: Router.00, (Local)
        Definition Equal to Local: Yes
        Disabled: No


Level-1:
        Definition Priority: 128
        Definition Source: Router.00, (Local)
        Definition Equal to Local: Yes
        Disabled: No


    Local Priority: 128
    FRR Disabled: No
    Microloop Avoidance Disabled: No
```

# Configuring SRv6 Locator Prefix Summarization

SRv6 leverages longest-prefix-match IP forwarding. Massive-scale reachability can be achieved by summarizing locators at ABRs and ASBRs.

Use the **summary-prefix** *locator* [**algorithm** *algo*] [**explicit**] command in IS-IS address-family configuration mode to specify that only locators from the specified algorithm contribute to the summary. The **explicit** keyword limits the contributing prefixes to only those belonging to the same algorithm.

The following example shows how to configure SRv6 IS-IS Algorithm Summarization for regular algorithm and Flexible Algorithm (128).

```
Router(config)# router isis core
Router(config-isis)# address-family ipv6 unicast
Router(config-isis-af)# summary-prefix 2001:0:1::/48
Router(config-isis-af)# summary-prefix 2001:0:2::/48 algorithm 128 explicit
```

# Configuring TI-LFA with SRv6 IS-IS

This feature introduces support for implementing Topology-Independent Loop-Free Alternate (TI-LFA) using SRv6 IS-IS.

TI-LFA provides link 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. TI-LFA leverages the post-convergence path which is planned to carry the traffic and ensures link and node protection within 50 milliseconds. TI-LFA with IS-IS SR-MPLS is already supported.

TI-LFA provides link, node, and Shared Risk Link Groups (SRLG) protection in any topology.

For more information, see Configure Topology-Independent Loop-Free Alternate (TI-LFA), on page 371.

### Usage Guidelines and Limitations

The following usage guidelines and limitations apply:

- TI-LFA provides link protection by default. Additional tiebreaker configuration is required to enable node or SRLG protection.
- Usage guidelines for node and SRLG protection:
  - TI-LFA node protection functionality 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.

- 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.

### Configuring SRv6 IS-IS TI-LFA

The following example shows how to configure different types of TI-LFA protection for SRv6 IS-IS.

```
Router(config)# router isis core
Router(config-isis)# interface bundle-ether 1201
Router(config-isis-if)# address-family ipv6 unicast
Router(config-isis-if-af)# fast-reroute per-prefix
Router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
Router(config-isis-if-af)# exit
Router(config-isis-if)# exit
Router(config-isis)# interface bundle-ether 1301
Router(config-isis-if)# address-family ipv6 unicast
Router(config-isis-if-af)# fast-reroute per-prefix
Router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
Router(config-isis-if-af)# fast-reroute per-prefix tiebreaker node-protecting index 100
Router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint index 200
Router(config-isis-if-af)# exit
```

### Configuring SRv6 IS-IS TI-LFA with Flexible Algorithm

TI-LFA backup paths for particular Flexible Algorithm are computed using the same constraints as the calculation of the primary paths for such Flexible Algorithm. These paths use the locator prefix advertised specifically for such Flexible Algorithm in order to enforce a backup path.

By default, LFA/TI-LFA for SRv6 Flexible Algorithm uses the LFA/TI-LFA configuration of Algo 0.

Use the **fast-reroute disable** command to disable the LFA/TI-LFA calculation on a per-algorithm basis:

```
Router(config)# router isis core
Router(config-isis)# flex-algo 128
Router(config-isis-flex-algo)# fast-reroute disable
```

### Verification

This example shows how to verify the SRv6 IS-IS TI-LFA configuration using the **show isis ipv6 fast-reroute** *ipv6-prefix* **detail** command.

```
Router# show isis ipv6 fast-reroute cafe:0:2::2/128 detail

L2 cafe:0:2::2/128 [20/115] Label: None, medium priority
     via fe80::e00:ff:fe3a:c700, HundredGigE0/0/0/0, Node2, Weight: 0
       Backup path: TI-LFA (link), via fe80::1600:ff:feec:fe00, HundredGigE0/0/0/1 Node3,
Weight: 0, Metric: 40
         P node: Node4.00 [cafe:0:4::4], SRv6 SID: cafe:0:4:: uN (PSP/USD)
         Backup-src: Node2.00
       P: No, TM: 40, LC: No, NP: No, D: No, SRLG: Yes
     src Node2.00-00, cafe:0:2::2
```

This example shows how to verify the SRv6 IS-IS TI-LFA configuration using the **show route ipv6** *ipv6-prefix* **detail** command.

```
Router# show route ipv6 cafe:0:2::2/128 detail
Tue Feb 23 23:08:48.151 UTC
```

```
Routing entry for cafe:0:2::2/128
  Known via "isis 1", distance 115, metric 20, type level-2
  Installed Feb 23 22:57:38.900 for 00:11:09
  Routing Descriptor Blocks
    fe80::1600:ff:feec:fe00, from cafe:0:2::2, via HundredGigE0/0/0/1, Backup (TI-LFA)
      Repair Node(s): cafe:0:4::4
      Route metric is 40
      Label: None
      Tunnel ID: None
      Binding Label: None
      Extended communities count: 0
      Path id:65              Path ref count:1
      NHID:0x20002(Ref:19)
      SRv6 Headend: H.Insert.Red [f3216], SID-list {cafe:0:4::}
    fe80::e00:ff:fe3a:c700, from cafe:0:2::2, via HundredGigE0/0/0/0, Protected
      Route metric is 20
      Label: None
      Tunnel ID: None
      Binding Label: None
      Extended communities count: 0
      Path id:1       Path ref count:0
      NHID:0x20001(Ref:19)
      Backup path id:65
  Route version is 0x4 (4)
  No local label
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
  Download Priority 1, Download Version 66
  No advertising protos.
```

This example shows how to verify the SRv6 IS-IS TI-LFA configuration using the **show cef ipv6** *ipv6-prefix* **detail location** *location* command.

```
Router# show cef ipv6 cafe:0:2::2/128 detail location 0/0/cpu0
Tue Feb 23 23:09:07.719 UTC
cafe:0:2::2/128, version 66, SRv6 Headend, internal 0x1000001 0x210 (ptr 0x8e96fd2c) [1],
0x0 (0x8e93fae0), 0x0 (0x8f7510a8)
 Updated Feb 23 22:57:38.904
 local adjacency to HundredGigE0/0/0/0

 Prefix Len 128, traffic index 0, precedence n/a, priority 1
  gateway array (0x8e7b5c78) reference count 1, flags 0x500000, source rib (7), 0 backups
                [2 type 3 flags 0x8401 (0x8e86ea40) ext 0x0 (0x0)]
  LW-LDI[type=3, refc=1, ptr=0x8e93fae0, sh-ldi=0x8e86ea40]
  gateway array update type-time 1 Feb 23 22:57:38.904
 LDI Update time Feb 23 22:57:38.913
 LW-LDI-TS Feb 23 22:57:38.913
   via fe80::1600:ff:feec:fe00/128, HundredGigE0/0/0/1, 9 dependencies, weight 0, class 0,
 backup (TI-LFA) [flags 0xb00]
    path-idx 0 NHID 0x20002 [0x8f5850b0 0x0]
    next hop fe80::1600:ff:feec:fe00/128, Repair Node(s): cafe:0:4::4
    local adjacency
    SRv6 H.Insert.Red SID-list {cafe:0:4::}
   via fe80::e00:ff:fe3a:c700/128, HundredGigE0/0/0/0, 6 dependencies, weight 0, class 0,
protected [flags 0x400]
    path-idx 1 bkup-idx 0 NHID 0x20001 [0x8f8420b0 0x0]
    next hop fe80::e00:ff:fe3a:c700/128

    Load distribution: 0 (refcount 2)
```

```
Hash  OK  Interface                Address
0     Y   HundredGigE0/0/0/0       fe80::e00:ff:fe3a:c700
```

# Configuring SRv6 IS-IS Microloop Avoidance

This feature introduces support for implementing microloop avoidance using IS-IS SRv6.

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 SRv6 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.

### Usage Guidelines and Limitations

The following usage guidelines and limitations apply:

- The Routing Information Base (RIB) update delay value specifies the amount of time the node uses the microloop avoidance policy before updating its forwarding table. The *delay-time* range is from 1 to 60000 milliseconds; the default value is 5000.

### Configuring SRv6 IS-IS Microloop Avoidance

The following example shows how to configure SRv6 IS-IS Microloop Avoidance and set the Routing Information Base (RIB) update delay value.

**Note** Complete the Configuring SRv6, on page 24 before performing these steps.

```
Router(config)# router isis test-igp
Router(config-isis)# address-family ipv6 unicast
Router(config-isis-af)# microloop avoidance segment-routing
Router(config-isis-af)# microloop avoidance rib-update-delay 2000
Router(config-isis-af)# commit
```

### Configuring SRv6 IS-IS Microloop Avoidance with Flexible Algorithm

Microloop Avoidance paths for particular Flexible Algorithm are computed using the same constraints as the calculation of the primary paths for such Flexible Algorithm. These paths use the Locator prefix advertised specifically for such Flexible Algorithm in order to enforce a microloop avoidance path.

By default, Microloop Avoidance for SRv6 Flexible Algorithm uses the Microloop Avoidance configuration of Algo 0.

Use the **microloop avoidance disable** command to disable the microloop calculation on a per-algorithm basis:

```
Router(config)# router isis test-tilfa
Router(config-isis)# flex-algo 128
Router(config-isis-flex-algo)# microloop avoidance disable
```

# Configuring SRv6 BGP-Based Services

Building on the messages and procedures defined in IETF draft "BGP/MPLS IP Virtual Private Networks (VPNs)", BGP has been extended to provide services over an SRv6 network, such as:

- IPv4 Layer-3 VPNs
- IPv6 Layer-3 VPNs
- IPv4 BGP global
- IPv6 BGP global
- Layer-2 VPNs - Ethernet VPNs (EVPN)

For more information about BGP, refer to the *BGP Configuration Guide for Cisco NCS 540 Series Routers* BGP Configuration Guide.

In SRv6-based services, the egress PE signals an SRv6 Service SID with the BGP service route. The ingress PE encapsulates the payload in an outer IPv6 header where the destination address is the SRv6 Service SID advertised by the egress PE. BGP messages between PEs carry SRv6 Service SIDs as a means to interconnect PEs and form VPNs. SRv6 Service SID refers to a segment identifier associated with one of the SRv6 service-specific behaviors advertised by the egress PE router, such as:

- uDT4 (Endpoint with decapsulation and IPv4 table lookup)
- uDT6 (Endpoint with decapsulation and IPv6 table lookup)
- uDX4 (Endpoint with decapsulation and IPv4 cross-connect)
- uDX6 (Endpoint with decapsulation and IPv6 cross-connect)

Based on the messages and procedures defined in IETF draft "SRv6 BGP based Overlay services", BGP encodes the SRv6 Service SID in the prefix-SID attribute of the corresponding BGP Network Layer Reachability Information (NLRI) and advertises it to its IPv6 BGP peers.

### Usage Guidelines and Restrictions

- The following SRv6 BGP-based services are supported:
  - IPv4 Layer-3 VPNs
  - IPv6 Layer-3 VPNs
  - IPv4 BGP global
  - IPv6 BGP global
- uDT4 and uDT6 for L3VPN and BGP global are supported.
- Dual-Stack L3 Services (IPv4 L3VPN, IPv6 L3VPN, IPv4 BGP global, IPv6 BGP global) are supported.

### SRv6 Locator Inheritance Rules

SRv6 locators can be assigned at different levels inside the BGP routing process. BGP allocates SRv6 Service SIDs from configured locator spaces according to the following inheritance rules:

1. Use the locator as defined under the service.

   If not defined under the specific service, then:

2. Use the locator as defined under the corresponding address-family.

   If not defined under the corresponding address-family, then:

3. Use the locator as defined globally under BGP.

### Enabling SRv6 Globally under BGP

Use the **router bgp** *as-number* **segment-routing srv6** command to enable SRv6 globally under the BGP routing process. The *as-number* is from 1-65535.

```
RP/0/0/CPU0:Node1(config)# router bgp 100 segment-routing srv6
```

### Assigning SRv6 Locator Globally under BGP

Use the **router bgp** *as-number* **segment-routing srv6 locator** *WORD* command to assign an SRv6 locator globally under the BGP routing process. The *as-number* is from 1-65535.

This example shows how to assign a locator:

```
RP/0/0/CPU0:Node1(config)# router bgp 100 segment-routing srv6 locator Node1-locator
```

For more information on how to configure an SRv6 locator, see Configuring SRv6, on page 24.

For more information on how to assign an SRv6 locator under the BGP service or BGP address-family, see the following SRv6 Services sections.

# SRv6 Services: IPv4 L3VPN

***Table 4: Feature History Table***

| Feature Name | Release | Description |
|---|---|---|
| Per-Prefix SRv6 Locator Assignment | Release 7.5.1 | This feature provides the ability to assign a specific SRv6 locator for a given prefix or a set of prefixes (IPv4/IPv6 GRT, IPv4/IPv6 VPN).<br>The egress PE advertises the prefix with the specified locator. This allows for per-prefix steering into desired transport behaviors, such as Flex Algo. |
| Support for iBGP as PE-CE protocol | Release 7.5.1 | This feature introduces support for iBGP as PE-CE protocol. |

| Feature Name | Release | Description |
|---|---|---|
| BGP Route Leaking | Release 7.5.1 | This feature adds support for importing routes from default-VRF to non-default VRF and routes from non-default VRF to default VRF. |

*Table 5: Feature History Table*

| Feature Name | Release | Description |
|---|---|---|
| Dual-Stack L3VPN Services (IPv4, IPv6) (SRv6 Micro-SID) | Release 7.3.2 | This feature introduces support for Dual-stack (VPNv4/VPNv6) VRFs.<br>VPNv4/VPNv6 Dual-stack supports both IPv4 (uDT4) and IPv6 (uDT6) based SRv6 L3VPN service on the same interface, sub-interface, or VRF. |

This feature provides IPv4 L3VPNs (VPNv4) over an SRv6 network.

### Usage Guidelines and Limitations

- SRv6 locator can be assigned globally, for all VRFs, or for an individual VRF.

  SRv6 locator can be assigned globally, for all VRFs, for an individual VRF, or per-prefix.
- Per-VRF allocation mode is supported (uDT4 behavior)
- Dual-Stack L3VPN Services (IPv4, IPv6) are supported
- Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
- eBGP, OSPF, Static are supported as PE-CE protocol.

  BGP (iBGP, eBGP), OSPF, Static are supported as PE-CE protocol.
- BGP route leaking between BGP Global and L3VPN is supported.
- MPLS L3VPN and SRv6 L3VPN interworking gateway is supported.
- Per-CE allocation mode is not supported (uDX4 behavior)
- iBGP is not supported as PE-CE protocol
- BGP route leaking is not supported

### Configuring SRv6 based IPv4 L3VPN

To enable SRv6-based L3VPN, you need to enable SRv6 under BGP, specify the locator, and configure the SID allocation mode. The assignment of the locator can be done in different places under the **router bgp** configuration. See SRv6 Locator Inheritance Rules, on page 39.

#### Use Case 1: Assigning SRv6 Locator Globally

This example shows how to enable SRv6 and configure the SRv6 locator name under BGP Global:

```
Node1(config)# router bgp 100
Node1(config-bgp)# segment-routing srv6
Node1(config-bgp-gbl-srv6)# locator Node1-locator
Node1(config-bgp-gbl-srv6)# exit
Node1(config-bgp)# address-family vpnv4 unicast
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 3001::1:1:1:4
Node1(config-bgp-nbr)# remote-as 100
Node1(config-bgp-nbr)# address-family vpnv4 unicast
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# vrf vrf_cust1
Node1(config-bgp-vrf)# rd 100:1
Node1(config-bgp-vrf)# address-family ipv4 unicast
Node1(config-bgp-vrf-af)# commit
```

### Running Config

```
router bgp 100
 segment-routing srv6
  locator Node1-locator
 !
 address-family vpnv4 unicast
 !
 neighbor 3001::1:1:1:4
  remote-as 100
  address-family vpnv4 unicast
  !
 !
 vrf vrf_cust1
  rd 100:1
  address-family ipv4 unicast
  !
 !
!
end
```

### Use Case 2: Assigning SRv6 Locator for All VRFs

To configure the SRv6 locator for all VRFs under VPNv4 Address Family and specify the allocation mode, use the following commands:

- **router bgp** *as-number* **address-family vpnv4 unicast vrf all segment-routing srv6**: Enable SRv6
- **router bgp** *as-number* **address-family vpnv4 unicast vrf all segment-routing srv6 alloc mode** {**per-vrf**}: Specify the SID behavior (allocation mode)
  - Use the **per-vrf** keyword to specify that the same service SID (uDT4 behavior) be used for all the routes advertised from a unique VRF.
- **router bgp** *as-number* **address-family vpnv4 unicast vrf all segment-routing srv6 locator** *WORD*: Specify the locator

This example shows how to enable SRv6 and configure the SRv6 locator for all VRFs under VPNv4 Address Family, with per-VRF label allocation mode:

```
Node1(config)# router bgp 100
Node1(config-bgp)# address-family vpnv4 unicast
Node1(config-bgp-af)# vrf all
Node1(config-bgp-af-vrfall)# segment-routing srv6
Node1(config-bgp-af-vrfall-srv6)# locator Node1-locator
```

```
Node1(config-bgp-af-vrfall-srv6)# alloc mode per-vrf
Node1(config-bgp-af-vrfall-srv6)# exit
Node1(config-bgp-af-vrfall)# exit
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 3001::1:1:1:4
Node1(config-bgp-nbr)# remote-as 100
Node1(config-bgp-nbr)# address-family vpnv4 unicast
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# vrf vrf_cust1
Node1(config-bgp-vrf)# rd 100:1
Node1(config-bgp-vrf)# address-family ipv4 unicast
Node1(config-bgp-vrf-af)# commit
```

### Running Config

```
router bgp 100
 address-family vpnv4 unicast
  vrf all
   segment-routing srv6
    locator Node1-locator
    alloc mode per-vrf
   !
  !
 !
 neighbor 3001::1:1:1:4
  remote-as 100
  address-family vpnv4 unicast
  !
 !
 vrf vrf_cust1
  rd 100:1
  address-family ipv4 unicast
  !
 !
!
end
```

### Use Case 3: Assigning SRv6 Locator for a specific VRF

To configure the SRv6 locator for a specific VRF under IPv4 Address Family and specify the allocation mode, use the following commands:

- **router bgp** *as-number* **vrf** *WORD* **address-family ipv4 unicast segment-routing srv6**: Enable SRv6
- **router bgp** *as-number* **vrf** *WORD* **address-family ipv4 unicast segment-routing srv6 alloc mode** { **per-vrf**}: Specify the SID behavior (allocation mode)
  - Use the **per-vrf** keyword to specify that the same service SID (uDT4 behavior) be used for all the routes advertised from a unique VRF.
- **router bgp** *as-number* **vrf** *WORD* **address-family ipv4 unicast segment-routing srv6 locator** *WORD*: Specify the locator

This example shows how to configure the SRv6 locator for an individual VRF, with per-VRF label allocation mode:

```
Node1(config)# router bgp 100
Node1(config-bgp)# address-family vpnv4 unicast
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 3001::1:1:1:4
```

```
Node1(config-bgp-nbr)# remote-as 100
Node1(config-bgp-nbr)# address-family vpnv4 unicast
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# vrf vrf_cust1
Node1(config-bgp-vrf)# rd 100:1
Node1(config-bgp-vrf)# address-family ipv4 unicast
Node1(config-bgp-vrf-af)# segment-routing srv6
Node1(config-bgp-vrf-af-srv6)# locator Node1-locator
Node1(config-bgp-vrf-af-srv6)# alloc mode per-vrf
Node1(config-bgp-vrf-af-srv6)# commit
```

### Running Config

```
router bgp 100
 address-family vpnv4 unicast
 !
 neighbor 3001::1:1:1:4
  remote-as 100
  address-family vpnv4 unicast
  !
 !
 vrf vrf_cust1
  rd 100:1
  address-family ipv4 unicast
   segment-routing srv6
    locator Node1-locator
    alloc mode per-vrf
   !
  !
 !
!
end
```

### Use Case 4: Assigning SRv6 Locator for a Specific Prefix

This use case provides the ability to assign a specific SRv6 locator for a given prefix or a set of prefixes. The egress PE advertises the prefix with the specified locator. This allows for per-prefix steering into desired transport behaviors, such as Flex Algo.

To assign an SRv6 locator for a specific prefix, configure a route policy to specify the SID allocation mode based on match criteria. Examples of match criteria are destination-based match or community-based match.

- Supported SID allocation modes are per-VRF and per-CE.
- For per-VRF allocation mode, you can also specify the SRv6 locator.
  - If an SRv6 locator is specified in the route policy, BGP will use that to allocate per-VRF SID. If the specified locator is invalid, the SID will not be allocated.
  - If an SRv6 locator is not specified in the route policy, the default locator configured under BGP is used to allocate the SID. If the default locator is not configured, then the SID will not be allocated.
- Per-CE allocation mode always uses the default locator configured under BGP to allocate the SID.

For more information on configuring routing policies, refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco NCS 540 Series Routers*.

The following example shows a route policy specifying the SID allocation mode with destination-based match:

```
Node1(config)# route-policy set_per_prefix_locator_rpl
Node1(config-rpl)# if destination in (10.1.1.0/24) then
```

```
Node1(config-rpl-if)# set srv6-alloc-mode per-vrf locator locator1
Node1(config-rpl-if)# elseif destination in (2.2.2.0/24) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf locator locator2
Node1(config-rpl-elseif)# elseif destination in (3.3.3.0/24) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf
Node1(config-rpl-elseif)# elseif destination in (4.4.4.0/24) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-ce
Node1(config-rpl-elseif)# else
Node1(config-rpl-else)# drop
Node1(config-rpl-else)# endif
Node1(config-rpl)# end-policy
Node1(config)#
```

To specify per-prefix allocation mode for a specific VRF under IPv4 address family, use the following command:

- **router bgp** *as-number* **vrf** *WORD* **address-family ipv4 unicast segment-routing srv6 alloc mode route-policy** *policy_name*

This example shows how to configure per-prefix allocation mode for a specific VRF (vrf_cust1) under IPv4 address family

```
Node1(config)# router bgp 100
Node1(config-bgp)# vrf vrf_cust1
Node1(config-bgp-vrf)# address-family ipv4 unicast
Node1(config-bgp-vrf-af)# segment-routing srv6
Node1(config-bgp-vrf-af-srv6)# alloc mode route-policy set_per_prefix_locator_rpl
```

### Running Configuration

```
route-policy set_per_prefix_locator_rpl
  if destination in (10.1.1.0/24) then
    set srv6-alloc-mode per-vrf locator locator1
  elseif destination in (2.2.2.0/24) then
    set srv6-alloc-mode per-vrf locator locator2
  elseif destination in (3.3.3.0/24) then
    set srv6-alloc-mode per-vrf
  elseif destination in (4.4.4.0/24) then
    set srv6-alloc-mode per-ce
  else
    drop
  endif
end-policy
!
router bgp 100
 vrf vrf_cust1
  address-family ipv6 unicast
   segment-routing srv6
    alloc mode route-policy set_per_prefix_locator_rpl
   !
  !
 !
!
```

Verify that the local and received SIDs have been correctly allocated under VPNv4 and specific VRF (vrf_cust1):

```
Node1# show bgp vpnv4 unicast local-sids
BGP router identifier 10.1.1.1, local AS number 100
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 50
BGP NSR Initial initsync version 18 (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            Local Sid                                 Alloc mode   Locator
Route Distinguisher: 8:8
*>i8.8.8.8/32         NO SRv6 Sid                               -            -
* i                   NO SRv6 Sid                               -            -
Route Distinguisher: 10.1.1.1:0 (default for vrf vrf_cust1)
*> 10.1.1.0/24        fc00:0:1:40::                             per-vrf      locator1
*> 2.2.2.0/24         fc00:8:1:40::                             per-vrf      locator2
*> 3.3.3.0/24         fc00:9:1:40::                             per-vrf      locator4
*> 4.4.4.0/24         fc00:9:1:41::                             per-ce       locator4
*> 10.1.1.5/32        NO SRv6 Sid                               -            -
*> 3.3.3.3/32         NO SRv6 Sid                               -            -
*>i8.8.8.8/32         NO SRv6 Sid                               -            -
```

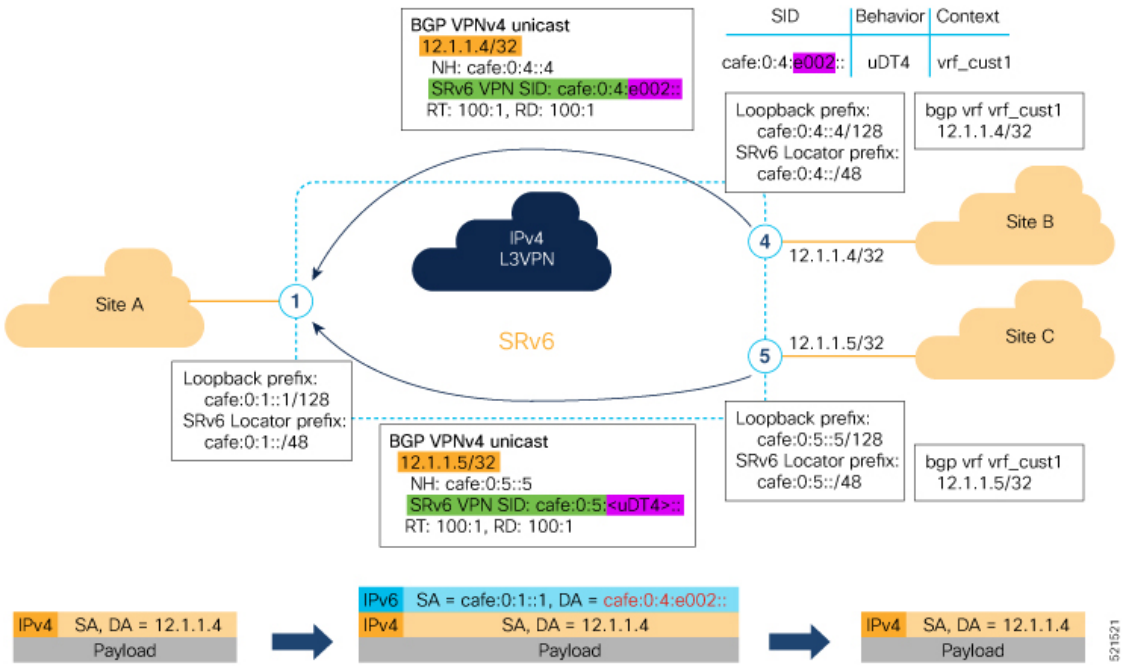
```
Node1# show bgp vpnv4 unicast received-sids
BGP router identifier 10.1.1.1, local AS number 100
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0   RD version: 0
BGP main routing table version 50
BGP NSR Initial initsync version 18 (Reached)
BGP NSR/ISSU 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                            Received Sid
Route Distinguisher: 8:8
*>i8.8.8.8/32         10.1.1.2                            fc00:0:2:42::
* i                   2400:2020:42:2fff::1
                                                          fc00:0:2:42::
Route Distinguisher: 10.1.1.1:0 (default for vrf vrf_cust1)
*> 10.1.1.0/24        11.1.1.2                            NO SRv6 Sid
*> 2.2.2.0/24         11.1.1.2                            NO SRv6 Sid
*> 3.3.3.0/24         11.1.1.2                            NO SRv6 Sid
*> 4.4.4.0/24         11.1.1.2                            NO SRv6 Sid
*> 10.1.1.5/32        11.1.1.2                            NO SRv6 Sid
*> 3.3.3.3/32         13.2.2.2                            NO SRv6 Sid
*>i8.8.8.8/32         10.1.1.2                            fc00:0:2:42::
```

```
Node1# show bgp vrf vrf_cust1 local-sids
BGP VRF vrf_cust1, state: Active
BGP Route Distinguisher: 10.1.1.1:0
VRF ID: 0x60000004
BGP router identifier 10.1.1.1, local AS number 1
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000013   RD version: 37
BGP main routing table version 37
BGP NSR Initial initsync version 18 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
```

```
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            Local Sid                                 Alloc mode   Locator
Route Distinguisher: 10.1.1.1:0 (default for vrf vrf_cust1)
*> 10.1.1.0/24        fc00:0:1:40::                             per-vrf      locator1
*> 2.2.2.0/24         fc00:8:1:40::                             per-vrf      locator2
*> 3.3.3.0/24         fc00:9:1:40::                             per-vrf      locator4
*> 4.4.4.0/24         fc00:9:1:41::                             per-ce       locator4
*> 10.1.1.5/32        NO SRv6 Sid                               -            -
*> 3.3.3.3/32         NO SRv6 Sid                               -            -
*>i8.8.8.8/32         NO SRv6 Sid                               -            -

Node1# show bgp vrf vrf_cust1 received-sids
BGP VRF vrf_cust1, state: Active
BGP Route Distinguisher: 10.1.1.1:0
VRF ID: 0x60000004
BGP router identifier 10.1.1.1, local AS number 1
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000013   RD version: 37
BGP main routing table version 37
BGP NSR Initial initsync version 18 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

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                            Received Sid
Route Distinguisher: 10.1.1.1:0 (default for vrf vrf_cust1)
*> 10.1.1.0/24        11.1.1.2                            NO SRv6 Sid
*> 2.2.2.0/24         11.1.1.2                            NO SRv6 Sid
*> 3.3.3.0/24         11.1.1.2                            NO SRv6 Sid
*> 4.4.4.0/24         11.1.1.2                            NO SRv6 Sid
*> 10.1.1.5/32        11.1.1.2                            NO SRv6 Sid
*> 3.3.3.3/32         13.2.2.2                            NO SRv6 Sid
*>i8.8.8.8/32         10.1.1.2                            fc00:0:2:42::
```

## Verification

The following figure shows a VPNv4 scenario. The sequence of commands included correspond to router Node1 acting as Ingress PE, and routers Node4 and Node5 acting as Egress PEs.




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

In this example, we can observe the uDT4 SIDs associated with the IPv4 L3VPN; where uDT4 behavior represents Endpoint with decapsulation and IPv4 table lookup.

```
Node1# show segment-routing srv6 sid

*** Locator: 'Node1-locator' ***

SID                         Behavior          Context                           Owner
          State  RW
--------------------------  ----------------  ------------------------------
------------------  -----  --
cafe:0:1::                  uN (PSP/USD)      'default':1                       sidmgr
          InUse  Y
cafe:0:1:e000::             uA (PSP/USD)      [Hu0/0/0/0, Link-Local]:0         isis-1
          InUse  Y
cafe:0:1:e001::             uA (PSP/USD)      [Hu0/0/0/1, Link-Local]:0         isis-1
          InUse  Y
cafe:0:1:e002::             uDT4              'vrf_cust1'                       bgp-100
          InUse  Y
cafe:0:1:e003::             uDT4              'vrf_cust2'                       bgp-100
          InUse  Y
cafe:0:1:e004::             uDT4              'vrf_cust3'                       bgp-100
          InUse  Y
cafe:0:1:e005::             uDT4              'vrf_cust4'                       bgp-100
          InUse  Y
cafe:0:1:e006::             uDT4              'vrf_cust5'                       bgp-100
          InUse  Y
```

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

```
Node1# show segment-routing srv6 sid cafe:0:1:e002:: detail
Tue Feb  9 17:50:40.621 UTC
```

```
*** Locator: 'Node1-locator' ***

SID                         Behavior          Context                                 Owner
               State  RW
--------------------------  ----------------  ------------------------------
------------------  -----  --
cafe:0:1:e002::             uDT4              'vrf_cust1'                             bgp-100
             InUse  Y
  SID Function: 0xe002
  SID context: { table-id=0xe0000011 ('vrf_cust1':IPv4/Unicast) }
  Locator: 'Node1-locator'
  Allocation type: Dynamic
  Created: Feb  9 17:41:07.475 (00:09:33 ago)
```

The following example shows how to verify the SRv6 based L3VPN configuration using the **show bgp vpnv4 unicast** commands on Egress PE.

```
Node1# show bgp vpnv4 unicast summary

BGP router identifier 10.1.1.1, local AS number 100
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 36
BGP NSR Initial initsync version 16 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

BGP is operating in STANDALONE mode.


Process       RcvTblVer   bRIB/RIB   LabelVer  ImportVer  SendTblVer  StandbyVer
Speaker              36         36         36         36          36           0

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down  St/PfxRcd
cafe:0:4::4       0   100      47      48       36    0    0 00:40:05          5
cafe:0:5::5       0   100      47      47       36    0    0 00:39:56          5


Node1# show bgp vpnv4 unicast rd 100:1

BGP router identifier 10.1.1.1, local AS number 100
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 36
BGP NSR Initial initsync version 16 (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: 100:1 (default for vrf vrf_cust1)
*> 12.1.1.1/32        0.0.0.0                  0         32768 ?
*>i12.4.4.4/32        cafe:0:4::4              0    100      0 ?
*>i12.5.5.5/32        cafe:0:5::5              0    100      0 ?

Processed 3 prefixes, 3 paths
```

```
Node1# show bgp vpnv4 unicast rd 100:1 12.4.4.4/32

BGP routing table entry for 12.4.4.4/32, Route Distinguisher: 100:1
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                22          22
Last Modified: Feb 23 22:57:56.756 for 00:40:08
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    cafe:0:4::4 (metric 30) from cafe:0:4::4 (10.1.1.4)
      Received Label 0xe00400
      Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best,
import-candidate, imported
      Received Path ID 0, Local Path ID 1, version 22
      Extended community: RT:1:1 RT:100:1
      PSID-Type:L3, SubTLV Count:1
       SubTLV:
        T:1(Sid information), Sid:cafe:0:4::, Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: VPNv4 Unicast, Source VRF: vrf_cust1, Source Route Distinguisher: 100:1
```

The following examples show how to verify the BGP prefix information for VRF instances using the **show bgp vrf** commands:

```
Node1# show bgp vrf vrf_cust1 ipv4 unicast

BGP VRF vrf_cust1, state: Active
BGP Route Distinguisher: 100:1
VRF ID: 0x60000002
BGP router identifier 10.1.1.1, local AS number 100
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000011   RD version: 32
BGP main routing table version 36
BGP NSR Initial initsync version 16 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

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: 100:1 (default for vrf vrf_cust1)
*> 12.1.1.1/32        0.0.0.0                  0         32768 ?
*>i12.4.4.4/32        cafe:0:4::4              0    100      0 ?
*>i12.5.5.5/32        cafe:0:5::5              0    100      0 ?

Processed 3 prefixes, 3 paths


Node1# show bgp vrf vrf_cust1 ipv4 unicast 12.4.4.4/32
Tue Feb 23 23:39:57.499 UTC
BGP routing table entry for 12.4.4.4/32, Route Distinguisher: 100:1
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                22          22
Last Modified: Feb 23 22:57:56.756 for 00:42:01
Paths: (1 available, best #1)
```

```
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    cafe:0:4::4 (metric 30) from cafe:0:4::4 (10.1.1.4)
      Received Label 0xe00400
      Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best,
import-candidate, imported
      Received Path ID 0, Local Path ID 1, version 22
      Extended community: RT:1:1 RT:100:1
      PSID-Type:L3, SubTLV Count:1
       SubTLV:
        T:1(Sid information), Sid:cafe:0:4::, Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: VPNv4 Unicast, Source VRF: vrf_cust1, Source Route Distinguisher: 100:1
```

The following example shows how to verify the SRv6 based L3VPN configuration using the **show route vrf** commands.

```
Node1# show route vrf vrf_cust1

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, t - Traffic Engineering, (!) - FRR Backup path

Gateway of last resort is not set

L    12.1.1.1/32 is directly connected, 00:44:43, Loopback100
B    12.4.4.4/32 [200/0] via cafe:0:4::4 (nexthop in vrf default), 00:42:45
B    12.5.5.5/32 [200/0] via cafe:0:5::5 (nexthop in vrf default), 00:42:45


Node1# show route vrf vrf_cust1 12.4.4.4/32

Routing entry for 12.4.4.4/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Feb 23 22:57:56.746 for 00:43:12
  Routing Descriptor Blocks
    cafe:0:4::4, from cafe:0:4::4
      Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
      Route metric is 0
  No advertising protos.


Node1# show route vrf vrf_cust1 12.4.4.4/32 detail

Routing entry for 12.4.4.4/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Feb 23 22:57:56.746 for 00:43:37
  Routing Descriptor Blocks
    cafe:0:4::4, from cafe:0:4::4
      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: 0x0000:100:1
    NHID:0x0(Ref:0)
    SRv6 Headend: H.Encaps.Red [f3216], SID-list {cafe:0:4:e004::}
  Route version is 0x1 (1)
  No local label
  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_REMOTE
  Download Priority 3, Download Version 3
  No advertising protos.
```

The following example shows how to verify the SRv6 based L3VPN configuration using the **show cef vrf** commands.

```
Node1# show cef vrf vrf_cust1

Prefix              Next Hop            Interface
------------------- ------------------- ------------------
0.0.0.0/0           drop                default handler
0.0.0.0/32          broadcast
12.1.1.1/32         receive             Loopback100
12.4.4.4/32         cafe:0:4::/128      <recursive>
12.5.5.5/32         cafe:0:5::/128      <recursive>
224.0.0.0/4         0.0.0.0/32
224.0.0.0/24        receive
255.255.255.255/32  broadcast


Node1# show cef vrf vrf_cust1 12.4.4.4/32

12.4.4.4/32, version 3, SRv6 Headend, internal 0x5000001 0x30 (ptr 0x78b9a61c) [1], 0x0
(0x0), 0x0 (0x88873720)
 Updated Feb 23 22:57:56.749
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
   via cafe:0:4::/128, 3 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x78e2da14 0x0]
    next hop VRF - 'default', table - 0xe0800000
    next hop cafe:0:4::/128 via cafe:0:4::/48
    SRv6 H.Encaps.Red SID-list {cafe:0:4:e004::}


Node1# show cef vrf vrf_cust1 12.4.4.4/32 detail

12.4.4.4/32, version 3, SRv6 Headend, internal 0x5000001 0x30 (ptr 0x78b9a61c) [1], 0x0
(0x0), 0x0 (0x88873720)
 Updated Feb 23 22:57:56.749
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
  gateway array (0x88a740a8) reference count 5, flags 0x2010, source rib (7), 0 backups
                [1 type 3 flags 0x48441 (0x789cbcc8) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Feb 23 22:57:56.749
 LDI Update time Feb 23 22:57:56.754

  Level 1 - Load distribution: 0
  [0] via cafe:0:4::/128, recursive

   via cafe:0:4::/128, 3 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x78e2da14 0x0]
    next hop VRF - 'default', table - 0xe0800000
    next hop cafe:0:4::/128 via cafe:0:4::/48
```

```
SRv6 H.Encaps.Red SID-list {cafe:0:4:e004::}

Load distribution: 0 1 (refcount 1)

Hash  OK  Interface                 Address
0     Y   HundredGigE0/0/0/1        remote
1     Y   HundredGigE0/0/0/0        remote
```

# SRv6 Services: IPv6 L3VPN

***Table 6: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SRv6 Services: IPv6 L3VPN | Release 7.3.1 | With this feature, the egress PE can signal an SRv6 Service SID with the BGP overlay service route. The ingress PE encapsulates the IPv4/IPv6 payload in an outer IPv6 header where the destination address is the SRv6 Service SID provided by the egress PE. BGP messages between PEs carry SRv6 Service SIDs as a means to interconnect PEs and form VPNs. |

This feature provides IPv6 L3VPNs (VPNv6) over an SRv6 network.

### Usage Guidelines and Limitations

- SRv6 locator can be assigned globally, for all VRFs, or for an individual VRF.

  SRv6 locator can be assigned globally, for all VRFs, for an individual VRF, or per-prefix.
- Per-VRF allocation mode is supported (uDT6 behavior)
- Dual-Stack L3VPN Services (IPv4, IPv6) are supported
- Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
- eBGP, OSPF, Static are supported as PE-CE protocol.

  BGP (iBGP, eBGP), OSPF, Static are supported as PE-CE protocol.
- BGP route leaking between BGP Global and L3VPN is supported.
- MPLS L3VPN and SRv6 L3VPN interworking gateway is supported.
- Per-CE allocation mode is not supported (uDX6 behavior)
- iBGP is not supported as PE-CE protocol
- BGP route leaking is not supported

### Configuring SRv6-based IPv6 L3VPN

To enable SRv6-based L3VPN, you need to enable SRv6 under BGP, specify the locator, and configure the SID allocation mode. The assignment of the locator can be done in different places under the **router bgp** configuration. See SRv6 Locator Inheritance Rules, on page 39.

#### Use Case 1: Assigning SRv6 Locator Globally

This example shows how to configure the SRv6 locator name under BGP Global:

```
Node1(config)# router bgp 100
Node1(config-bgp)# segment-routing srv6
Node1(config-bgp-gbl-srv6)# locator Node1-locator
Node1(config-bgp-gbl-srv6)# exit
Node1(config-bgp)# address-family vpnv6 unicast
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 3001::12:1:1:4
Node1(config-bgp-nbr)# remote-as 100
Node1(config-bgp-nbr)# address-family vpnv6 unicast
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# vrf vrf_cust6
Node1(config-bgp-vrf)# rd 100:6
Node1(config-bgp-vrf)# address-family ipv6 unicast
Node1(config-bgp-vrf-af)# commit
```

#### Running Configuration

```
router bgp 100
 segment-routing srv6
  locator Node1-locator
 !
 address-family vpnv6 unicast
 !
 neighbor 3001::12:1:1:4
  remote-as 100
  address-family vpnv6 unicast
  !
 !
 vrf vrf_cust6
  rd 100:6
  address-family ipv6 unicast
  !
 !
!
end
```

#### Use Case 2: Assigning SRv6 Locator for All VRFs

To configure the SRv6 locator for all VRFs under VPNv6 Address Family and specify the allocation mode, use the following commands:

- **router bgp** *as-number* **address-family vpnv6 unicast vrf all segment-routing srv6**: Enable SRv6
- **router bgp** *as-number* **address-family vpnv6 unicast vrf all segment-routing srv6 alloc mode** {**per-vrf**}: Specify the SID behavior (allocation mode)
  - Use the **per-vrf** keyword to specify that the same service SID (uDT6 behavior) be used for all the routes advertised from a unique VRF.
- **router bgp** *as-number* **address-family vpnv6 unicast vrf all segment-routing srv6 locator** *WORD*: Specify the locator

This example shows how to configure the SRv6 locator for all VRFs under VPNv6 Address Family, with per-VRF label allocation mode:

```
Node1(config)# router bgp 100
Node1(config-bgp)# address-family vpnv6 unicast
Node1(config-bgp-af)# vrf all
Node1(config-bgp-af-vrfall)# segment-routing srv6
Node1(config-bgp-af-vrfall-srv6)# locator Node1-locator
Node1(config-bgp-af-vrfall-srv6)# alloc mode per-vrf
Node1(config-bgp-af-vrfall-srv6)# exit
Node1(config-bgp-af-vrfall)# exit
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 3001::12:1:1:4
Node1(config-bgp-nbr)# remote-as 100
Node1(config-bgp-nbr)# address-family vpnv6 unicast
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# vrf vrf_cust6
Node1(config-bgp-vrf)# rd 100:6
Node1(config-bgp-vrf)# address-family ipv6 unicast
Node1(config-bgp-vrf-af)# commit
```

### Running Configuration

```
router bgp 100
 address-family vpnv6 unicast
  vrf all
   segment-routing srv6
    locator Node1-locator
    alloc mode per-vrf
   !
  !
 !
 neighbor 3001::12:1:1:4
  remote-as 100
  address-family vpnv6 unicast
  !
 !
 vrf vrf_cust6
  rd 100:6
  address-family ipv6 unicast
  !
 !
!
end
```

### Use Case 3: Assigning SRv6 Locator for a specific VRF

To configure the SRv6 locator for a specific VRF under IPv6 Address Family and specify the allocation mode, use the following commands:

- **router bgp** *as-number* **vrf** *WORD* **address-family ipv6 unicast segment-routing srv6**: Enable SRv6
- **router bgp** *as-number* **vrf** *WORD* **address-family ipv6 unicast segment-routing srv6 alloc mode** { **per-vrf**}: Specify the SID behavior (allocation mode)
  - Use the **per-vrf** keyword to specify that the same service SID (uDT6 behavior) be used for all the routes advertised from a unique VRF.
- **router bgp** *as-number* **vrf** *WORD* **address-family ipv6 unicast segment-routing srv6 locator** *WORD*: Specify the locator

This example shows how to configure the SRv6 locator for an individual VRF, with per-VRF label allocation mode:

```
Node1(config)# router bgp 100
Node1(config-bgp)# address-family vpnv6 unicast
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 3001::12:1:1:4
Node1(config-bgp-nbr)# remote-as 100
Node1(config-bgp-nbr)# address-family vpnv6 unicast
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# vrf vrf_cust6
Node1(config-bgp-vrf)# rd 100:6
Node1(config-bgp-vrf)# address-family ipv6 unicast
Node1(config-bgp-vrf-af)# segment-routing srv6
Node1(config-bgp-vrf-af-srv6)# locator Node1-locator
Node1(config-bgp-vrf-af-srv6)# alloc mode per-vrf
Node1(config-bgp-vrf-af-srv6)# commit
```

### Running Configuration

```
router bgp 100
 address-family vpnv6 unicast
 !
 neighbor 3001::12:1:1:4
  remote-as 100
  address-family vpnv6 unicast
  !
 !
 vrf vrf_cust6
  rd 100:6
  address-family ipv6 unicast
   segment-routing srv6
    locator Node1-locator
    alloc mode per-vrf
   !
  !
 !
!
end
```

### Use Case 4: Assigning SRv6 Locator for a Specific Prefix

This use case provides the ability to assign a specific SRv6 locator for a given prefix or a set of prefixes. The egress PE advertises the prefix with the specified locator. This allows for per-prefix steering into desired transport behaviors, such as Flex Algo.

To assign an SRv6 locator for a specific prefix, configure a route policy to specify the SID allocation mode based on match criteria. Examples of match criteria are destination-based match or community-based match.

- Supported SID allocation modes are per-VRF and per-CE.
- For per-VRF allocation mode, you can also specify the SRv6 locator.
  - If an SRv6 locator is specified in the route policy, BGP will use that to allocate per-VRF SID. If the specified locator is invalid, the SID will not be allocated.
  - If an SRv6 locator is not specified in the route policy, the default locator is used to allocate the SID. If the default locator is not configured in BGP, then the SID will not be allocated.
- Per-CE allocation mode always uses the default locator configured under BGP to allocate the SID.

For more information on configuring routing policies, refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide Cisco NCS 540 Series Routers*.

The following example shows a route policy specifying the SID allocation mode with destination-based match:

```
Node1(config)# route-policy set_per_prefix_locator_rpl
Node1(config-rpl)# if destination in (3001::1:1:1:1/128) then
Node1(config-rpl-if)# set srv6-alloc-mode per-vrf locator locator1
Node1(config-rpl-if)# elseif destination in (3001::2:2:2:2/128) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf locator locator2
Node1(config-rpl-elseif)# elseif destination in (3001::3:3:3:3/128) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf
Node1(config-rpl-elseif)# elseif destination in (3001::4:4:4:4/128) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-ce
Node1(config-rpl-elseif)# else
Node1(config-rpl-else)# drop
Node1(config-rpl-else)# endif
Node1(config-rpl)# end-policy
```

To specify per-prefix allocation mode for a specific VRF under IPv6 Address Family, use the following command:

- **router bgp** *as-number* **vrf** *WORD* **address-family ipv6 unicast segment-routing srv6 alloc mode route-policy** *policy_name*

This example shows how to specify per-prefix allocation mode for a specific VRF (vrf_cust6) under the IPv6 address family:

```
Node1(config)# router bgp 100
Node1(config-bgp)# vrf vrf_cust6
Node1(config-bgp-vrf)# address-family ipv6 unicast
Node1(config-bgp-vrf-af)# segment-routing srv6
Node1(config-bgp-vrf-af-srv6)# alloc mode route-policy set_per_prefix_locator_rpl
```

### Running Configuration

```
route-policy set_per_prefix_locator_rpl
  if destination in (3001::1:1:1:1/128) then
    set srv6-alloc-mode per-vrf locator locator1
  elseif destination in (3001::2:2:2:2/128) then
    set srv6-alloc-mode per-vrf locator locator2
  elseif destination in (3001::3:3:3:3/128) then
    set srv6-alloc-mode per-vrf
  elseif destination in (3001::4:4:4:4/128) then
    set srv6-alloc-mode per-ce
  else
    drop
  endif
end-policy
!
router bgp 100
 vrf vrf_cust6
  address-family ipv6 unicast
   segment-routing srv6
    alloc mode route-policy set_per_prefix_locator_rpl
   !
  !
 !
!
```

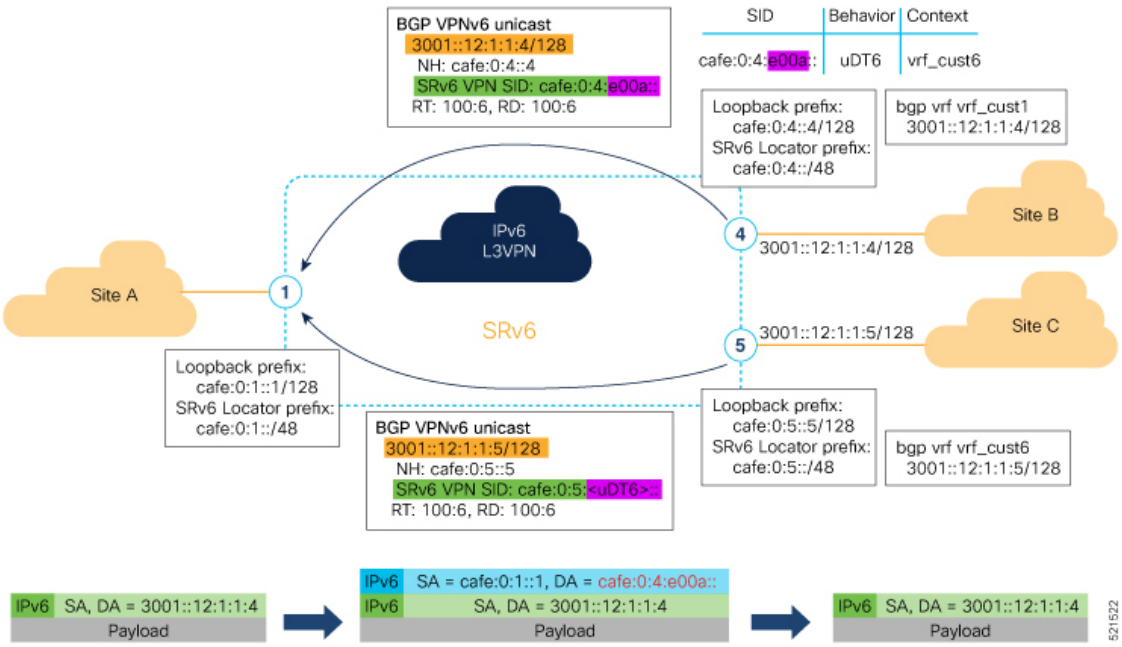
Verify that the local and received SIDs have been correctly allocated under VPNv6 and specific VRF (vrf_cust6):

```
Node1# show bgp vpnv6 unicast local-sids

BGP router identifier 10.1.1.1, local AS number 1
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 50
BGP NSR Initial initsync version 18 (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            Local Sid                                Alloc mode   Locator
Route Distinguisher: 8:8
*>i3008::8:8:8:8/128  NO SRv6 Sid                              -            -
* i                   NO SRv6 Sid                              -            -
Route Distinguisher: 100:6 (default for vrf vrf_cust6)
*> 3001::1:1:1:1/128  fc00:0:1:40::                            per-vrf      locator1
*> 3001::2:2:2:2/128  fc00:8:1:40::                            per-vrf      locator2
*> 3001::3:3:3:3/128  fc00:9:1:40::                            per-vrf      locator4
*> 3001::4:4:4:4/128  fc00:9:1:41::                            per-ce       locator4
*> 3001::5:5:5:5/128  NO SRv6 Sid                              -            -
*> 3001::12:1:1:5/128 NO SRv6 Sid                              -            -
*>i3008::8:8:8:8/128  NO SRv6 Sid                              -            -


Node1# show bgp vpnv6 unicast received-sids

BGP router identifier 10.1.1.1, local AS number 1
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 50
BGP NSR Initial initsync version 18 (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                            Received Sid
Route Distinguisher: 8:8
*>i3008::8:8:8:8/128  10.1.1.2                            fc00:0:2:42::
* i                   2400:2020:42:2fff::1
                                                          fc00:0:2:42::
Route Distinguisher: 100:6 (default for vrf vrf_cust6)
*> 3001::1:1:1:1/128  11.1.1.2                            NO SRv6 Sid
*> 3001::2:2:2:2/128  11.1.1.2                            NO SRv6 Sid
*> 3001::3:3:3:3/128  11.1.1.2                            NO SRv6 Sid
*> 3001::4:4:4:4/128  11.1.1.2                            NO SRv6 Sid
*> 3001::5:5:5:5/128  11.1.1.2                            NO SRv6 Sid
*> 3001::12:1:1:5/128 13.2.2.2                            NO SRv6 Sid
*>i3008::8:8:8:8/128  10.1.1.2                            fc00:0:2:42::


Node1# show bgp vrf vrf_cust6 local-sids
BGP VRF vrf_cust6, state: Active
BGP Route Distinguisher: 10.1.1.1:0
```

```
VRF ID: 0x60000004
BGP router identifier 10.1.1.1, local AS number 1
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000013   RD version: 37
BGP main routing table version 37
BGP NSR Initial initsync version 18 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

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            Local Sid                                 Alloc mode   Locator
Route Distinguisher: 8:8
*>i3008::8:8:8:8/128  NO SRv6 Sid                               -            -
* i                   NO SRv6 Sid                               -            -
Route Distinguisher: 100:6 (default for vrf vrf_cust6)
*> 3001::1:1:1:1/128  fc00:0:1:40::                             per-vrf      locator1
*> 3001::2:2:2:2/128  fc00:8:1:40::                             per-vrf      locator2
*> 3001::3:3:3:3/128  fc00:9:1:40::                             per-vrf      locator4
*> 3001::4:4:4:4/128  fc00:9:1:41::                             per-ce       locator4
*> 3001::5:5:5:5/128  NO SRv6 Sid                               -            -
*> 3001::12:1:1:5/128 NO SRv6 Sid                               -            -
*>i3008::8:8:8:8/128  NO SRv6 Sid                               -            -


Node1# show bgp vrf vrf_cust6 received-sids
BGP VRF vrf_cust6, state: Active
BGP Route Distinguisher: 100:6
VRF ID: 0x60000004
BGP router identifier 10.1.1.1, local AS number 1
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000013   RD version: 37
BGP main routing table version 37
BGP NSR Initial initsync version 18 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

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                            Received Sid
Route Distinguisher: 100:6 (default for vrf vrf_cust6)
*> 3001::1:1:1:1/128  11.1.1.2                            NO SRv6 Sid
*> 3001::2:2:2:2/128  11.1.1.2                            NO SRv6 Sid
*> 3001::3:3:3:3/128  11.1.1.2                            NO SRv6 Sid
*> 3001::4:4:4:4/128  11.1.1.2                            NO SRv6 Sid
*> 3001::5:5:5:5/128  11.1.1.2                            NO SRv6 Sid
*> 3001::12:1:1:5/128 13.2.2.2                            NO SRv6 Sid
*>i3008::8:8:8:8/128  10.1.1.2                            fc00:0:2:42::
```

### Verification

The following figure shows a VPNv6 scenario. The sequence of commands included correspond to router Node1 acting as Ingress PE, and routers Node4 and Node5 acting as Egress PEs.




The following examples shows how to verify the SRv6 based L3VPN configurations for an Individual VRF with per VRF label allocation mode.

In this example, we can observe the uDT6 SID associated with the IPv6 L3VPN, where uDT6 behavior represents Endpoint with decapsulation and IPv6 table lookup.

```
Node1# show segment-routing srv6 sid
Fri Jan 29 19:31:53.293 UTC

*** Locator: 'Node1-locator' ***

SID                         Behavior          Context                          Owner
          State  RW
--------------------------  ----------------  -------------------------------
------------------  -----  --
cafe:0:1::                  uN (PSP/USD)      'default':1                      sidmgr
          InUse  Y
cafe:0:1:e000::             uA (PSP/USD)      [Hu0/0/0/0, Link-Local]:0        isis-1
          InUse  Y
cafe:0:1:e001::             uA (PSP/USD)      [Hu0/0/0/1, Link-Local]:0        isis-1
          InUse  Y
cafe:0:1:e002::             uDT4              'vrf_cust1'                      bgp-100
          InUse  Y
cafe:0:1:e003::             uDT4              'vrf_cust2'                      bgp-100
          InUse  Y
cafe:0:1:e004::             uDT4              'vrf_cust3'                      bgp-100
          InUse  Y
cafe:0:1:e005::             uDT4              'vrf_cust4'                      bgp-100
          InUse  Y
cafe:0:1:e006::             uDT4              'vrf_cust5'                      bgp-100
          InUse  Y
cafe:0:1:e007::             uA (PSP/USD)      [Hu0/0/0/0, Link-Local]:0:P      isis-1
          InUse  Y
cafe:0:1:e008::             uA (PSP/USD)      [Hu0/0/0/1, Link-Local]:0:P      isis-1
          InUse  Y
cafe:0:1:e009::             uDT6              'default'                        bgp-100
          InUse  Y
cafe:0:1:e00a::             uDT6              'vrf_cust6'                      bgp-100
```

```
          InUse  Y
```

The following examples show how to verify the SRv6 based L3VPN configuration using the **show bgp vpnv6 unicast** commands on the Ingress PE.

```
Node1# show bgp vpnv6 unicast summary
Fri Jan 29 19:33:01.177 UTC
BGP router identifier 10.1.1.1, local AS number 100
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 6
BGP NSR Initial initsync version 4 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

BGP is operating in STANDALONE mode.


Process       RcvTblVer   bRIB/RIB   LabelVer  ImportVer  SendTblVer  StandbyVer
Speaker               6          6          6          6           6           0

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down  St/PfxRcd
cafe:0:4::4       0   100     122     123        6    0    0 00:20:05          1
cafe:0:5::5       0   100     111     111        0    0    0 00:49:46          1

Node1# show bgp vpnv6 unicast rd 100:6
Fri Jan 29 19:41:01.334 UTC
BGP router identifier 10.1.1.1, local AS number 100
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 8
BGP NSR Initial initsync version 4 (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: 100:6 (default for vrf vrf_cust6)
*> 3001::12:1:1:1/128 ::                       0         32768 ?
*>i3001::12:1:1:4/128 cafe:0:4::4              0    100      0 ?
*>i3001::12:1:1:5/128 cafe:0:5::5              0    100      0 ?

Processed 3 prefixes, 3 paths

Node1# show bgp vpnv6 unicast rd 100:6 3001::12:1:1:4/128
Fri Jan 29 19:41:42.008 UTC
BGP routing table entry for 3001::12:1:1:4/128, Route Distinguisher: 100:6
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                  6           6
Last Modified: Jan 29 19:29:35.858 for 00:12:06
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    cafe:0:4::4 (metric 30) from cafe:0:4::4 (10.1.1.4)
```

```
      Received Label 0xe00a00
      Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best,
import-candidate, imported
      Received Path ID 0, Local Path ID 1, version 6
      Extended community: RT:100:6
      PSID-Type:L3, SubTLV Count:1
       SubTLV:
        T:1(Sid information), Sid:cafe:0:4::, Behavior:62, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: VPNv6 Unicast, Source VRF: vrf_cust6, Source Route Distinguisher: 100:6
```

The following examples show how to verify the BGP prefix information for VRF instances:

```
Node1# show bgp vrf vrf_cust6 ipv6 unicast
Fri Jan 29 19:42:05.675 UTC
BGP VRF vrf_cust6, state: Active
BGP Route Distinguisher: 100:6
VRF ID: 0x60000007
BGP router identifier 10.1.1.1, local AS number 100
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0800016   RD version: 8
BGP main routing table version 8
BGP NSR Initial initsync version 4 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

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: 100:6 (default for vrf vrf_cust6)
*> 3001::12:1:1:1/128 ::                       0         32768 ?
*>i3001::12:1:1:4/128 cafe:0:4::4              0    100      0 ?
*>i3001::12:1:1:5/128 cafe:0:5::5              0    100      0 ?

Processed 3 prefixes, 3 paths

Node1# show bgp vrf vrf_cust6 ipv6 unicast 3001::12:1:1:4/128

BGP routing table entry for 3001::12:1:1:4/128, Route Distinguisher: 100:6
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 17          17
Last Modified: Jan 15 16:50:44.032 for 01:48:21
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    cafe:0:4::4 (metric 30) from cafe:0:4::4 (10.1.1.4)
      Received Label 0xe00a00
      Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best,
import-candidate, imported
      Received Path ID 0, Local Path ID 1, version 17
      Extended community: RT:100:6
      PSID-Type:L3, SubTLV Count:1
       SubTLV:
        T:1(Sid information), Sid:cafe:0:4::, Behavior:62, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: VPNv6 Unicast, Source VRF: vrf_cust6, Source Route Distinguisher: 100:6
```

The following examples show how to verify the current routes in the Routing Information Base (RIB):

```
Node1# show route vrf vrf_cust6 ipv6 unicast
Fri Jan 29 19:43:28.067 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, t - Traffic Engineering, (!) - FRR Backup path

Gateway of last resort is not set

L    3001::12:1:1:1/128 is directly connected,
      01:01:23, Loopback105
B    3001::12:1:1:4/128
      [200/0] via cafe:0:4::4 (nexthop in vrf default), 00:13:52
B    3001::12:1:1:5/128
      [200/0] via cafe:0:5::5 (nexthop in vrf default), 00:05:53

Node1# show route vrf vrf_cust6 ipv6 unicast 3001::12:1:1:4/128
Fri Jan 29 19:43:55.645 UTC

Routing entry for 3001::12:1:1:4/128
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Jan 29 19:29:35.696 for 00:14:20
  Routing Descriptor Blocks
    cafe:0:4::4, from cafe:0:4::4
      Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
      Route metric is 0
  No advertising protos.

Node1# show route vrf vrf_cust6 ipv6 unicast 3001::12:1:1:4/128 detail
Fri Jan 29 19:44:17.914 UTC

Routing entry for 3001::12:1:1:4/128
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Jan 29 19:29:35.696 for 00:14:42
  Routing Descriptor Blocks
    cafe:0:4::4, from cafe:0:4::4
      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: 0x0000:100:6
      NHID:0x0(Ref:0)
      SRv6 Headend: H.Encaps.Red [f3216], SID-list {cafe:0:4:e00a::}
  Route version is 0x1 (1)
  No local label
  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_REMOTE
  Download Priority 3, Download Version 3
  No advertising protos.
```

The following examples show how to verify the current IPv6 Cisco Express Forwarding (CEF) table:

```
Node1# show cef vrf vrf_cust6 ipv6
Fri Jan 29 19:44:56.888 UTC

::/0
  drop       default handler
3001::12:1:1:1/128
  receive    Loopback105
3001::12:1:1:4/128
  recursive    cafe:0:4::/128
3001::12:1:1:5/128
  recursive    cafe:0:5::/128
fe80::/10
  receive
ff02::/16
  receive
ff02::2/128
  receive
ff02::1:ff00:0/104
  receive
ff05::/16
  receive
ff12::/16
  receive

Node1# show cef vrf vrf_cust6 ipv6 3001::12:1:1:4/128
Fri Jan 29 19:45:23.607 UTC
3001::12:1:1:4/128, version 3, SRv6 Headend, internal 0x5000001 0x30 (ptr 0x78f2e0e0) [1],
 0x0 (0x0), 0x0 (0x888a3ac8)
 Updated Jan 29 19:29:35.700
 Prefix Len 128, traffic index 0, precedence n/a, priority 3
   via cafe:0:4::/128, 7 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x78cd2a14 0x0]
    next hop VRF - 'default', table - 0xe0800000
    next hop cafe:0:4::/128 via cafe:0:4::/48
    SRv6 H.Encaps.Red SID-list {cafe:0:4:e00a::}

Node1# show cef vrf vrf_cust6 ipv6 3001::12:1:1:4/128 detail
Fri Jan 29 19:45:55.847 UTC
3001::12:1:1:4/128, version 3, SRv6 Headend, internal 0x5000001 0x30 (ptr 0x78f2e0e0) [1],
 0x0 (0x0), 0x0 (0x888a3ac8)
 Updated Jan 29 19:29:35.700
 Prefix Len 128, traffic index 0, precedence n/a, priority 3
  gateway array (0x78afe238) reference count 1, flags 0x2010, source rib (7), 0 backups
               [1 type 3 flags 0x48441 (0x78ba9a60) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Jan 29 19:29:35.699
 LDI Update time Jan 29 19:29:35.701

  Level 1 - Load distribution: 0
  [0] via cafe:0:4::/128, recursive

   via cafe:0:4::/128, 7 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x78cd2a14 0x0]
    next hop VRF - 'default', table - 0xe0800000
    next hop cafe:0:4::/128 via cafe:0:4::/48
    SRv6 H.Encaps.Red SID-list {cafe:0:4:e00a::}

    Load distribution: 0 1 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   HundredGigE0/0/0/0        remote
```

```
1      Y    HundredGigE0/0/0/1         remote
```

# SRv6 Services: IPv4 BGP Global

This feature extends support of SRv6-based BGP services to include IPv4 global BGP by implementing uDT4 SRv6 functions at the PE node (draft-ietf-bess-srv6-services).

### Usage Guidelines and Limitations

- SRv6 locator can be assigned globally or under IPv4 unicast address family
- Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
- BGP, OSPF, Static are supported as PE-CE protocol.
- BGP route leaking between BGP Global and L3VPN is supported.
- Dual-Stack L3 Services (IPv4 BGP global, IPv6 BGP global) are supported.

### BGP Global IPv4 Over SRv6 with Per-AFI SID Allocation Mode (uDT4)

To configure BGP global IPv4 over SRv6, use the following commands:

- **router bgp** *as-number* **address-family ipv4 unicast segment-routing srv6**: Enable SRv6
- **router bgp** *as-number* **address-family ipv4 unicast segment-routing srv6 alloc mode per-vrf**: Specify the SID behavior (allocation mode).

  The **per-vrf** keyword specifies that the same label is be used for all the routes advertised from a unique VRF.
- **router bgp** *as-number* **address-family ipv4 unicast segment-routing srv6 alloc mode** {**per-vrf** | **route-policy** *policy_name*}: Specify the SID behavior (allocation mode).
  - **per-vrf**: Specifies that the same label is be used for all the routes advertised from a unique VRF.
  - **route-policy** *policy_name*: Uses a route policy to determine the SID allocation mode and locator (if provided) for given prefix.
- **router bgp** *as-number* **address-family ipv4 unicast segment-routing srv6 locator** *WORD*: Specify the locator
- **router bgp** *as-number* {**af-group** *WORD*| **neighbor-group** *WORD* | **neighbor** *ipv6-addr*} **address-family ipv4 unicast encapsulation-type srv6**: Specify the encapuslation type for SRv6.
  - Use **af-group** *WORD* to apply the SRv6 encapsulation type to the address family group for BGP neighbors.
  - Use **neighbor-group** *WORD*to apply the SRv6 encapsulation type to the neighbor group for BGP neighbors.
  - Use **neighbor** *ipv6-addr* to apply the SRv6 encapsulation type to the specific BGP neighbor.

#### Use Case 1: BGP Global IPv4 over SRv6 with Per-AFI SID Allocation

The following example shows how to configure BGP global IPv4 over SRv6 with per-AFI SID allocation.

```
Node1(config)# router bgp 1
Node1(config-bgp)# bgp router-id 10.1.0.1
Node1(config-bgp)# address-family ipv4 unicast
Node1(config-bgp-af)# segment-routing srv6
Node1(config-bgp-af-srv6)# locator Node1
Node1(config-bgp-af-srv6)# alloc mode per-vrf
Node1(config-bgp-af-srv6)# exit
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor 60::2
Node1(config-bgp-nbr)# remote-as 1
Node1(config-bgp-nbr)# update-source Loopback1
Node1(config-bgp-nbr)# address-family ipv4 unicast
Node1(config-bgp-nbr-af)# encapsulation-type srv6
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# neighbor 52.52.52.1
Node1(config-bgp-nbr)# remote-as 3
Node1(config-bgp-nbr)# address-family ipv4 unicast
Node1(config-bgp-nbr-af)# route-policy passall in
Node1(config-bgp-nbr-af)# route-policy passall out
Node1(config-bgp-nbr-af)# commit
```

### Running Configuration

```
router bgp 1
 bgp router-id 10.1.0.1
 address-family ipv4 unicast
  segment-routing srv6
   locator Node1
   alloc mode per-vrf
  !
 !
 neighbor 60::2
  remote-as 1
  update-source Loopback1
  address-family ipv4 unicast
   encapsulation-type srv6
  !
 !
 neighbor 52.52.52.1
  remote-as 3
  address-family ipv4 unicast
   route-policy passall in
   route-policy passall out
  !
 !
!
```

### Use Case 2: BGP Global IPv4 over SRv6 with Per-Prefix SID Allocation

This use case provides the ability to assign a specific SRv6 locator for a given prefix or a set of prefixes. The egress PE advertises the prefix with the specified locator. This allows for per-prefix steering into desired transport behaviors, such as Flex Algo.

To assign an SRv6 locator for a specific prefix, configure a route policy to specify the SID allocation mode based on match criteria. Examples of match criteria are destination-based match or community-based match.

- Supported SID allocation modes are per-VRF and per-CE.
- For per-VRF allocation mode, you can also specify the SRv6 locator.

- If an SRv6 locator is specified in the route policy, BGP will use that to allocate per-VRF SID. If the specified locator is invalid, the SID will not be allocated.
- If an SRv6 locator is not specified in the route policy, the default locator is used to allocate the SID. If the default locator is not configured in BGP, then the SID will not be allocated.

- Per-CE allocation mode always uses the default locator to allocate the SID.

For more information on configuring routing policies, refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco NCS 540 Series Routers*.

The following example shows a route policy specifying the SID allocation mode with destination-based match:

```
Node1(config)# route-policy set_per_prefix_locator_rpl
Node1(config-rpl)# if destination in (10.1.1.0/24) then
Node1(config-rpl-if)# set srv6-alloc-mode per-vrf locator locator1
Node1(config-rpl-if)# elseif destination in (2.2.2.0/24) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf locator locator2
Node1(config-rpl-elseif)# elseif destination in (3.3.3.0/24) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf
Node1(config-rpl-elseif)# elseif destination in (4.4.4.0/24) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-ce
Node1(config-rpl-elseif)# else
Node1(config-rpl-else)# drop
Node1(config-rpl-else)# endif
Node1(config-rpl)# end-policy
Node1(config)#
```

The following example shows how to configure BGP global IPv4 over SRv6 with a route policy to determine the SID allocation mode for given prefix.

```
Node1(config)# router bgp 100
Node1(config-bgp)# address-family ipv4 unicast
Node1(config-bgp-af)# segment-routing srv6
Node1(config-bgp-af-srv6)# alloc mode route-policy set_per_prefix_locator_rpl
```

### Running Configuration

```
route-policy set_per_prefix_locator_rpl
  if destination in (10.1.1.0/24) then
    set srv6-alloc-mode per-vrf locator locator1
  elseif destination in (2.2.2.0/24) then
    set srv6-alloc-mode per-vrf locator locator2
  elseif destination in (3.3.3.0/24) then
    set srv6-alloc-mode per-vrf
  elseif destination in (4.4.4.0/24) then
    set srv6-alloc-mode per-ce
  else
    drop
  endif
end-policy
!
router bgp 100
 address-family ipv4 unicast
  segment-routing srv6
   alloc mode route-policy set_per_prefix_locator_rpl
  !
 !
!
```

Verify that the local and received SIDs have been correctly allocated under BGP IPv4 address family:

```
Node1# show bgp ipv4 unicast local-sids
…
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            Local Sid                                Alloc mode   Locator
*> 10.1.1.0/24        fc00:8:1:41::                            per-vrf      locator2
*> 2.2.2.0/24         fc00:0:1:41::                            per-vrf      locator1
*> 3.3.3.0/24         fc00:9:1:42::                            per-vrf      locator4
*> 4.4.4.0/24         fc00:9:1:43::                            per-ce       locator4
*> 10.1.1.5/32        NO SRv6 Sid                              -            -
* i8.8.8.8/32         NO SRv6 Sid                              -            -


Node1# show bgp ipv4 unicast received-sids
…
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                          Received Sid
*> 10.1.1.0/24        66.2.2.2                          NO SRv6 Sid
*> 2.2.2.0/24         66.2.2.2                          NO SRv6 Sid
*> 3.3.3.0/24         66.2.2.2                          NO SRv6 Sid
*> 4.4.4.0/24         66.2.2.2                          NO SRv6 Sid
*> 10.1.1.5/32        66.2.2.2                          NO SRv6 Sid
* i8.8.8.8/32         77.1.1.2                          fc00:0:2:41::
```

# SRv6 Services: IPv6 BGP Global

***Table 7: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SRv6 Services: BGP Global IPv6 | Release 7.3.1 | With this feature, the egress PE can signal an SRv6 Service SID with the BGP global route. The ingress PE encapsulates the IPv4/IPv6 payload in an outer IPv6 header where the destination address is the SRv6 Service SID provided by the egress PE. BGP messages between PEs carry SRv6 Service SIDs as a means to interconnect PEs. |

This feature extends support of SRv6-based BGP services to include IPv6 global BGP by implementing uDT6 SRv6 functions at the PE node (draft-ietf-bess-srv6-services).

### Usage Guidelines and Limitations

- SRv6 locator can be assigned globally or under IPv6 unicast address family
- Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
- BGP, OSPF, Static are supported as PE-CE protocol.

- BGP route leaking between BGP Global and L3VPN is supported.
- Dual-Stack L3 Services (IPv4 BGP global, IPv6 BGP global) are supported.

### BGP Global IPv6 Over SRv6 with Per-AFI SID Allocation Mode (uDT6)

To configure BGP global IPv6 over SRv6, use the following commands:

- **router bgp** *as-number* **address-family ipv6 unicast segment-routing srv6**: Enable SRv6
- **router bgp** *as-number* **address-family ipv6 unicast segment-routing srv6 alloc mode per-vrf**: Specify the SID behavior (allocation mode).

  The **per-vrf** keyword specifies that the same label is be used for all the routes advertised from a unique VRF.
- **router bgp** *as-number* **address-family ipv6 unicast segment-routing srv6 alloc mode** {**per-vrf** | **route-policy** *policy_name*}: Specify the SID behavior (allocation mode).
  - **per-vrf**: Specifies that the same label is be used for all the routes advertised from a unique VRF.
  - **route-policy** *policy_name*: Uses a route policy to determine the SID allocation mode and locator (if provided) for given prefix.
- **router bgp** *as-number* **address-family ipv6 unicast segment-routing srv6 locator** *WORD*: Specify the locator
- **router bgp** *as-number* {**af-group** *WORD*| **neighbor-group** *WORD* | **neighbor** *ipv6-addr*} **address-family ipv6 unicast encapsulation-type srv6**: Specify the encapuslation type for SRv6.
  - Use **af-group** *WORD* to apply the SRv6 encapsulation type to the address family group for BGP neighbors.
  - Use **neighbor-group** *WORD*to apply the SRv6 encapsulation type to the neighbor group for Border Gateway Protocol (BGP) neighbors.
  - Use **neighbor** *ipv6-addr* to apply the SRv6 encapsulation type to the specific BGP neighbor.

#### Use Case 1: BGP Global IPv6 over SRv6 with Per-AFI SID Allocation

The following example shows how to configure BGP global IPv6 over SRv6 with per-AFI SID allocation.

```
Node1(config)# router bgp 100
Node1(config-bgp)# bgp router-id 10.1.1.1
Node1(config-bgp)# segment-routing srv6
Node1(config-bgp-gbl-srv6)# locator Node1
Node1(config-bgp-gbl-srv6)# exit
Node1(config-bgp)# address-family ipv6 unicast
Node1(config-bgp-af)# segment-routing srv6
Node1(config-bgp-af-srv6)# locator Node1
Node1(config-bgp-af-srv6)# alloc mode per-vrf
Node1(config-bgp-af-srv6)# exit
Node1(config-bgp-af)# exit
Node1(config-bgp)# neighbor cafe:0:4::4
Node1(config-bgp-nbr)# address-family ipv6 unicast
Node1(config-bgp-nbr-af)# encapsulation-type srv6
Node1(config-bgp-nbr-af)# exit
Node1(config-bgp-nbr)# exit
Node1(config-bgp)# neighbor cafe:0:5::5
```

```
Node1(config-bgp-nbr)# address-family ipv6 unicast
Node1(config-bgp-nbr-af)# encapsulation-type srv6
Node1(config-bgp-nbr-af)# commit
```

### Running Configuration

```
router bgp 100
 bgp router-id 10.1.1.1
 segment-routing srv6
  locator Node1
 !
 address-family ipv6 unicast
  segment-routing srv6
   locator Node1
   alloc mode per-vrf
  !
 !
 neighbor cafe:0:4::4
  address-family ipv6 unicast
   encapsulation-type srv6
  !
 !
 neighbor cafe:0:5::5
  address-family ipv6 unicast
   encapsulation-type srv6
```

### Use Case 2: BGP Global IPv6 over SRv6 with Per-Prefix SID Allocation

This use case provides the ability to assign a specific SRv6 locator for a given prefix or a set of prefixes. The egress PE advertises the prefix with the specified locator. This allows for per-prefix steering into desired transport behaviors, such as Flex Algo.

To assign an SRv6 locator for a specific prefix, configure a route policy to specify the SID allocation mode based on match criteria. Examples of match criteria are destination-based match or community-based match.

- Supported SID allocation modes are per-VRF and per-CE.
- For per-VRF allocation mode, you can also specify the SRv6 locator.
  - If an SRv6 locator is specified in the route policy, BGP will use that to allocate per-VRF SID. If the specified locator is invalid, the SID will not be allocated.
  - If an SRv6 locator is not specified in the route policy, the default locator is used to allocate the SID. If the default locator is not configured in BGP, then the SID will not be allocated.
- Per-CE allocation mode always uses the default locator to allocate the SID.

For more information on configuring routing policies, refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco NCS 540 Series Routers*.

The following example shows a route policy specifying the SID allocation mode with destination-based match:

```
Node1(config)# route-policy set_per_prefix_locator_rpl
Node1(config-rpl)# if destination in (3001::1:1:1:1/128) then
Node1(config-rpl-if)# set srv6-alloc-mode per-vrf locator locator1
Node1(config-rpl-if)# elseif destination in (3001::2:2:2:2/128) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf locator locator2
Node1(config-rpl-elseif)# elseif destination in (3001::3:3:3:3/128) then
Node1(config-rpl-elseif)# set srv6-alloc-mode per-vrf
Node1(config-rpl-elseif)# elseif destination in (3001::4:4:4:4/128) then
```

```
Node1(config-rpl-elseif)# set srv6-alloc-mode per-ce
Node1(config-rpl-elseif)# else
Node1(config-rpl-else)# drop
Node1(config-rpl-else)# endif
Node1(config-rpl)# end-policy
```

The following example shows how to configure BGP global IPv6 over SRv6 with a route policy to determine the SID allocation mode for given prefix.

```
Node1(config)# router bgp 100
Node1(config-bgp)# address-family ipv6 unicast
Node1(config-bgp-af)# segment-routing srv6
Node1(config-bgp-af-srv6)# alloc mode route-policy set_per_prefix_locator_rpl
```

### Running Configuration

```
route-policy set_per_prefix_locator_rpl
  if destination in (3001::1:1:1:1/128) then
    set srv6-alloc-mode per-vrf locator locator1
  elseif destination in (3001::2:2:2:2/128) then
    set srv6-alloc-mode per-vrf locator locator2
  elseif destination in (3001::3:3:3:3/128) then
    set srv6-alloc-mode per-vrf
  elseif destination in (3001::4:4:4:4/128) then
    set srv6-alloc-mode per-ce
  else
    drop
  endif
end-policy
!
router bgp 100
 address-family ipv6 unicast
  segment-routing srv6
   alloc mode route-policy set_per_prefix_locator_rpl
  !
 !
```

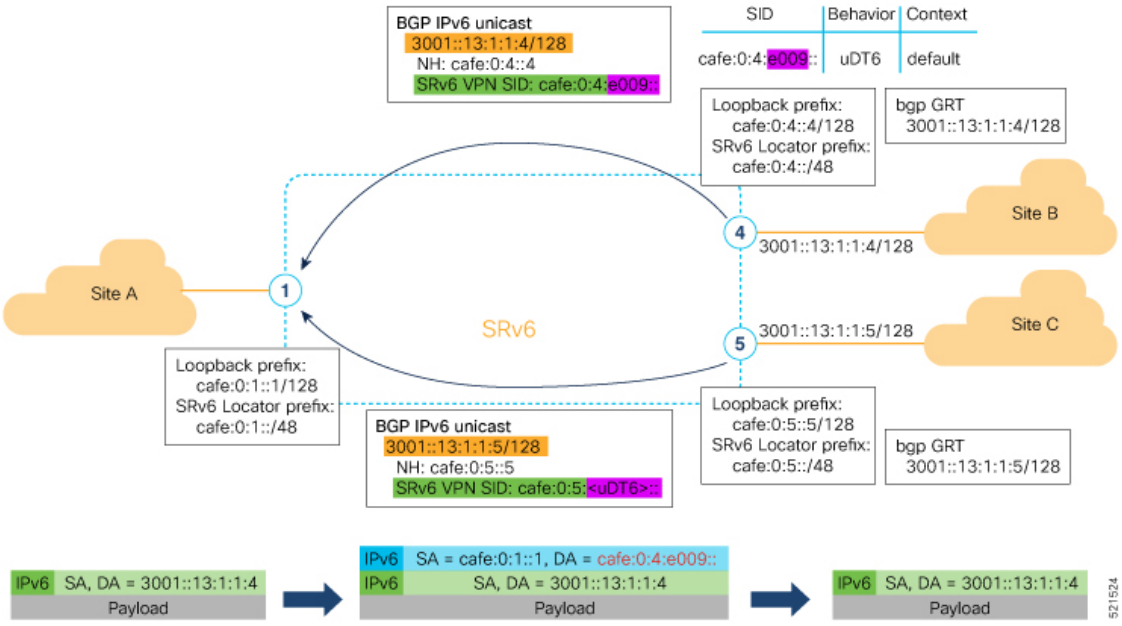
Verify that the local and received SIDs have been correctly allocated under BGP IPv6 address family:

```
Node1# show bgp ipv6 unicast local-sids
…
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            Local Sid                                Alloc mode   Locator
*> 3001::1:1:1:1/128  fc00:8:1:41::                            per-vrf      locator2
*> 3001::2:2:2:2/128  fc00:0:1:41::                            per-vrf      locator1
*> 3001::3:3:3:3/128  fc00:9:1:42::                            per-vrf      locator4
*> 3001::4:4:4:4/128  fc00:9:1:43::                            per-ce       locator4
*> 3001::5:5:5:5/128  NO SRv6 Sid                              -            -
* i3008::8:8:8:8/128  NO SRv6 Sid                              -            -


Node1# show bgp ipv6 unicast received-sids
…
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                            Received Sid
*> 3001::1:1:1:1/128  66.2.2.2                            NO SRv6 Sid
*> 3001::2:2:2:2/128  66.2.2.2                            NO SRv6 Sid
*> 3001::3:3:3:3/128  66.2.2.2                            NO SRv6 Sid
```

```
*> 3001::4:4:4:4/128   66.2.2.2                                NO SRv6 Sid
*> 3001::5:5:5:5/128   66.2.2.2                                NO SRv6 Sid
* i3008::8:8:8:8/128   77.1.1.2                                fc00:0:2:41::
```

## Verification

The following figure shows a IPv6 BGP global scenario. The sequence of commands included correspond to router Node1 acting as Ingress PE, and routers Node4 and Node5 acting as Egress PEs.




The following examples show how to verify the BGP global IPv6 configuration using the **show bgp ipv6 unicast** commands.

```
Node1# show bgp ipv6 unicast summary
Fri Jan 29 19:48:23.255 UTC
BGP router identifier 10.1.1.1, local AS number 100
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0800000   RD version: 4
BGP main routing table version 4
BGP NSR Initial initsync version 2 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

BGP is operating in STANDALONE mode.


Process       RcvTblVer   bRIB/RIB   LabelVer  ImportVer  SendTblVer  StandbyVer
Speaker               4          4          4          4           4           0

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down  St/PfxRcd
cafe:0:4::4       0   100     137     138        4    0    0 00:35:27          1
cafe:0:5::5       0   100     138     137        4    0    0 00:10:54          1

Node1# show bgp ipv6 unicast
Fri Jan 29 19:49:05.688 UTC
BGP router identifier 10.1.1.1, local AS number 100
```

```
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0800000   RD version: 4
BGP main routing table version 4
BGP NSR Initial initsync version 2 (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
*> 3001::13:1:1:1/128 ::                       0         32768 i
*>i3001::13:1:1:4/128 cafe:0:4::4              0    100      0 i
*>i3001::13:1:1:5/128 cafe:0:5::5              0    100      0 i

Processed 3 prefixes, 3 paths

Node1# show bgp ipv6 unicast 3001::13:1:1:4/128
Fri Jan 29 19:49:22.067 UTC
BGP routing table entry for 3001::13:1:1:4/128
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 3           3
Last Modified: Jan 29 19:14:13.858 for 00:35:08
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    cafe:0:4::4 (metric 30) from cafe:0:4::4 (10.1.1.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best, group-best
      Received Path ID 0, Local Path ID 1, version 3
      PSID-Type:L3, SubTLV Count:1
       SubTLV:
        T:1(Sid information), Sid:cafe:0:4:e009::, Behavior:62, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
```

The following examples show how to verify the current routes in the Routing Information Base (RIB):

```
Node1# show route ipv6 3001::13:1:1:4/128
Fri Jan 29 19:53:26.839 UTC

Routing entry for 3001::13:1:1:4/128
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Jan 29 19:14:13.397 for 00:35:28
  Routing Descriptor Blocks
    cafe:0:4::4, from cafe:0:4::4
      Route metric is 0
  No advertising protos.

Node1# show route ipv6 3001::13:1:1:4/128 detail
Fri Jan 29 19:50:08.601 UTC

Routing entry for 3001::13:1:1:4/128
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Jan 29 19:14:13.397 for 00:35:55
  Routing Descriptor Blocks
    cafe:0:4::4, from cafe:0:4::4
      Route metric is 0
      Label: None
```

```
    Tunnel ID: None
    Binding Label: None
    Extended communities count: 0
    NHID:0x0(Ref:0)
    SRv6 Headend: H.Encaps.Red [f3216], SID-list {cafe:0:4:e009::}
  Route version is 0x1 (1)
  No local label
  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 106
  No advertising protos.
```

The following examples show how to verify the current IPv6 Cisco Express Forwarding (CEF) table:

```
Node1# show cef ipv6 3001::13:1:1:4/128
Fri Jan 29 19:50:29.149 UTC
3001::13:1:1:4/128, version 106, SRv6 Headend, internal 0x5000001 0x40 (ptr 0x78    cd3944)
 [1], 0x0 (0x0), 0x0 (0x888a3a80)
 Updated Jan 29 19:14:13.401
 Prefix Len 128, traffic index 0, precedence n/a, priority 4
   via cafe:0:4::/128, 7 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x78cd2a14 0x0]
    next hop cafe:0:4::/128 via cafe:0:4::/48
    SRv6 H.Encaps.Red SID-list {cafe:0:4:e009::}

Node1# show cef ipv6 3001::13:1:1:4/128 detail
Fri Jan 29 19:51:00.920 UTC
3001::13:1:1:4/128, version 106, SRv6 Headend, internal 0x5000001 0x40 (ptr 0x78cd3944)
[1], 0x0 (0x0), 0x0 (0x888a3a80)
 Updated Jan 29 19:14:13.401
 Prefix Len 128, traffic index 0, precedence n/a, priority 4
  gateway array (0x78afe150) reference count 1, flags 0x2010, source rib (7), 0 backups
                [1 type 3 flags 0x48441 (0x78ba99e8) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Jan 29 19:14:13.401
 LDI Update time Jan 29 19:14:13.401

  Level 1 - Load distribution: 0
  [0] via cafe:0:4::/128, recursive

   via cafe:0:4::/128, 7 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x78cd2a14 0x0]
    next hop cafe:0:4::/128 via cafe:0:4::/48
    SRv6 H.Encaps.Red SID-list {cafe:0:4:e009::}

    Load distribution: 0 1 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   HundredGigE0/0/0/0        remote
    1     Y   HundredGigE0/0/0/1        remote
```

# SRv6 Services: IPv4 L3VPN Active-Standby Redundancy using Port-Active Mode

The Segment Routing IPv6 (SRv6) Services: IPv4 L3VPN 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.

### Usage Guidelines and 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
Router(config-evpn-ac-es)# identifier type 0 11.11.11.11.11.11.11.11.14
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 l2vpn 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 l2vpn evpn
Router(config-bgp-nbr)# commit
```

## Running Configuration

```
interface Bundle-Ether14
 ipv4 address 14.0.0.2 255.255.255.0
 ipv6 address 14::2/64
 lacp period short
 mac-address 1.2.3
 bundle wait-while 0
!
interface GigabitEthernet0/2/0/5
 bundle id 14 mode active
!
evpn
 interface Bundle-Ether14
  ethernet-segment
   identifier type 0 11.11.11.11.11.11.11.11.14
   load-balancing-mode port-active
  !
 !
!
router bgp 100
 bgp router-id 192.168.0.2
 address-family l2vpn evpn
 !
 neighbor 192.168.0.3
  remote-as 100
  update-source Loopback0
  address-family l2vpn evpn
  !
 !
!
```

## Verification

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

```
/* Verify ethernet-segment details on active DF router */
```

```
Router# show evpn ethernet-segment interface Bundle-Ether14 detail
Ethernet Segment Id      Interface                         Nexthops
------------------------ --------------------------------- --------------------
0011.1111.1111.1111.1114 BE14                              192.168.0.2
                                                           192.168.0.3
    ES to BGP Gates   : Ready
  ES to L2FIB Gates : Ready
  Main port         :
     Interface name : Bundle-Ether14
     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:
     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 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
```

# SRv6 Services: IPv4 L3VPN Active-Active Redundancy

This feature provides active-active connectivity to a CE device in a L3VPN deployment. The CE device can be Layer-2 or Layer-3 device connecting to the redundant PEs over a single LACP LAG port.

Depending on the bundle hashing, an ARP or IPv6 Network Discovery (ND) packet can be sent to any of the redundant routers. As a result, not all entries will exist on a given PE. In order to provide complete awareness, Layer-3 local route learning is augmented with remote route-synchronization programming.

Route synchronization between service PEs is required in order to provide minimum interruption to unicast and multicast services after failure on a redundant service PE. The following EVPN route-types are used for Layer-3 route synchronization:

- EVPN route-type 2 for synchronizing ARP tables
- EVPN route-type 7/8 for synchronizing IGMP JOINS/LEAVES

In a Layer-3 CE scenario, the router that connects to the redundant PEs may establish an IGP adjacency on the bundle port. In this case, the adjacency will be formed to one of the redundant PEs, and IGP customer routes will only be present on that PE. To synchronize Layer-3 customer subnet routes (IP Prefixes), the EVPN route-type 5 is used to carry the ESI and ETAG as well as the gateway address (prefix next-hop address).

**Note** Gratuitous ARP (GARP) or IPv6 Network Advertisement (NA) replay is not needed for CEs connected to the redundant PEs over a single LAG port.

The below configuration enables Layer-3 route synchronization for routes learned on the Ethernet-segment sub-interfaces.

```
evpn
 route-sync vrf default
!
vrf RED
 evi route-sync 10
!
vrf BLUE
 evi route-sync 20
!
```

**Note** EVPN does not support untagged interfaces.

# SRv6 Services: EVPN VPWS — All-Active Multi-Homing

*Table 8: Feature History Table*

| Feature Name | Release | Description |
|---|---|---|
| SRv6 Services: EVPN VPWS — All-Active Multi-Homing (SRv6 Micro SID) | Release 7.3.2 | This feature provides an ELINE (P2P) service with all-active multihoming capability over an SRv6 network.<br>All-Active Multi-Homing enables an operator to connect a customer edge (CE) device to two or more provider edge (PE) devices to provide load balancing and redundant connectivity. With All-Active Multi-Homing, all the PEs can forward traffic to and from the multi-homed device. |

EVPN VPWS All-Active Multi-Homing over SRv6 provides an ELINE (P2P) service with all-active multihoming capability over an SRv6 network.

All-Active Multi-Homing enables an operator to connect a customer edge (CE) device to two or more provider edge (PE) devices to provide load balancing and redundant connectivity. With All-Active Multi-Homing, all the PEs can forward traffic to and from the multi-homed device.

**Note** For information about EVPN VPWS, refer to the "EVPN Virtual Private Wire Service (VPWS)" chapter in the *L2VPN and Ethernet Services Configuration Guide for Cisco NCS 540 Series Routers*.

### Configuring EVPN VPWS over SRv6

**Note** Complete the steps in Configuring SRv6, on page 24 before performing these steps.

An SRv6 Locator for an EVPN VPWS service can be configured at 3 different levels independently:

- global_locator is the default locator for all EVPN-VPWS services
- evi_locator is applied to all EVPN-VPWS services for the specific EVI
- evi_service_locator is applied to an individual EVI service

When locators are configured at different levels at the same time, the following priority is implemented:

1. evi_service_locator
2. evi_locator

3. global_locator

This example shows how to configure an EVPN VPWS over SRv6 using a global locator for EVPN:

```
evpn
 segment-routing srv6
  locator sample_global_loc

l2vpn
 xconnect group sample_xcg
  p2p sample-vpws-12001-2002
   interface Bundle-Ether12001.2002
   neighbor evpn evi 12001 service 2002 segment-routing srv6
```

This example shows how to configure EVPN VPWS over SRv6 using specific EVI locator:

```
evpn
 evi 11001 segment-routing srv6
  locator sample_evi_loc

l2vpn
 xconnect group sample_xcg
  p2p sample-vpws-11001-2002
   interface Bundle-Ether11001.2002
   neighbor evpn evi 11001 service 2002 segment-routing srv6
```

This example shows how to configure an EVPN VPWS over SRv6 using a locator for an individual EVI service:

```
l2vpn
 xconnect group sample_xcg
  p2p sample-vpws-11001-2001
   interface Bundle-Ether11001.2001
   neighbor evpn evi 11001 service 2001 segment-routing srv6
    locator sample_evi_service_loc
```

## Verification

```
Router# show segment-routing srv6 locator

Name                  ID       Algo  Prefix                    Status   Flags
--------------------  -------  ----  ------------------------  -------  --------
sample_evi_loc        1        128   2001:0:8::/48             Up       U
sample_global_loc     2        0     2001:0:1::/48             Up       U


Router# show segment-routing srv6 sid

*** Locator: 'sample_evi_loc' ***

SID                         Behavior          Context                          Owner
          State  RW
--------------------------  ----------------  -------------------------------
------------------  -----  --
2001:0:8::                  uN (PSP/USD)      'default':8                      sidmgr
          InUse  Y
2001:0:8:e000::             uDX2              11001:2002                       l2vpn_srv6
          InUse  Y
2001:0:8:e002::             uA (PSP/USD)      [BE11, Link-Local]:128           isis-20
          InUse  Y
2001:0:8:e004::             uA (PSP/USD)      [BE60, Link-Local]:128           isis-20
```

```
          InUse  Y
2001:0:8:e006::             uA (PSP/USD)      [BE30, Link-Local]:128           isis-20
          InUse  Y

*** Locator: 'sample_global_loc' ***

2001:0:1::                  uN (PSP/USD)      'default':1                      sidmgr
          InUse  Y
2001:0:1:e001::             uDX2              12001:2002                       l2vpn_srv6
          InUse  Y
2001:0:1:e003::             uA (PSP/USD)      [BE11, Link-Local]:0             isis-20
          InUse  Y
2001:0:1:e005::             uA (PSP/USD)      [BE60, Link-Local]:0             isis-20
          InUse  Y
2001:0:1:e007::             uA (PSP/USD)      [BE30, Link-Local]:0             isis-20
          InUse  Y


Router# show evpn segment-routing srv6 detail

Configured default locator: sample_global_loc
EVIs with unknown locator config: 0
VPWS with unknown locator config: 0

Locator name         Prefix                      OOR      Service count    SID count
------------         ------                      ---      -------------    ---------
sample_evi_loc       2001:0:8::/48               False    1                1
   Configured on EVIs <evi>: 11001
sample_global_loc    2001:0:1::/48               False    1                1
   Default locator


Router# show l2vpn xconnect group sample_xcg detail
Thu Sep  2 14:39:22.575 UTC


Group sample_xcg, XC sample-vpws-11001-2002, state is up; Interworking none
  AC: Bundle-Ether11001.2002, state is up
    Type VLAN; Num Ranges: 1
    Rewrite Tags: []
    VLAN ranges: [2002, 2002]
    MTU 1504; XC ID 0xc0002ee8; interworking none
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
      drops: illegal VLAN 0, illegal length 0
  EVPN: neighbor ::ffff:10.0.0.1, PW ID: evi 11001, ac-id 2002, state is up ( established
)
    XC ID 0xa0001f47
    Encapsulation SRv6
    Encap type Ethernet
    Ignore MTU mismatch: Enabled
    Transmit MTU zero: Disabled
    Reachability: Up

      SRv6              Local                       Remote
      ----------------  --------------------------- --------------------------
      uDX2              2001:0:8:e000::             2001:0:3:e000::
      AC ID             2002                        2002
      MTU               1518                        1518
      Locator           sample_evi_loc              N/A
      Locator Resolved  Yes                         N/A
      SRv6 Headend      H.Encaps.L2.Red             N/A
```

```
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0

Group sample_xcg, XC sample-vpws-12001-2002, state is up; Interworking none
  AC: Bundle-Ether12001.2002, state is up
    Type VLAN; Num Ranges: 1
    Rewrite Tags: []
    VLAN ranges: [2002, 2002]
    MTU 1504; XC ID 0xc0002eea; interworking none
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
      drops: illegal VLAN 0, illegal length 0
  EVPN: neighbor ::ffff:10.0.0.2, PW ID: evi 12001, ac-id 2002, state is up ( established
)
    XC ID 0xa0001f49
    Encapsulation SRv6
    Encap type Ethernet
    Ignore MTU mismatch: Enabled
    Transmit MTU zero: Disabled
    Reachability: Up

      SRv6              Local                       Remote
      ----------------  --------------------------- --------------------------
      uDX2              2001:0:1:e001::             2001:0:2:e001::
      AC ID             2002                        2002
      MTU               1518                        1518
      Locator           sample_global_loc           N/A
      Locator Resolved  Yes                         N/A
      SRv6 Headend      H.Encaps.L2.Red             N/A
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
```

# SRv6-Services: EVPN ELAN Layer 2 Gateway With Automated Steering To Flexible Algorithm Paths

*Table 9: Feature History Table*

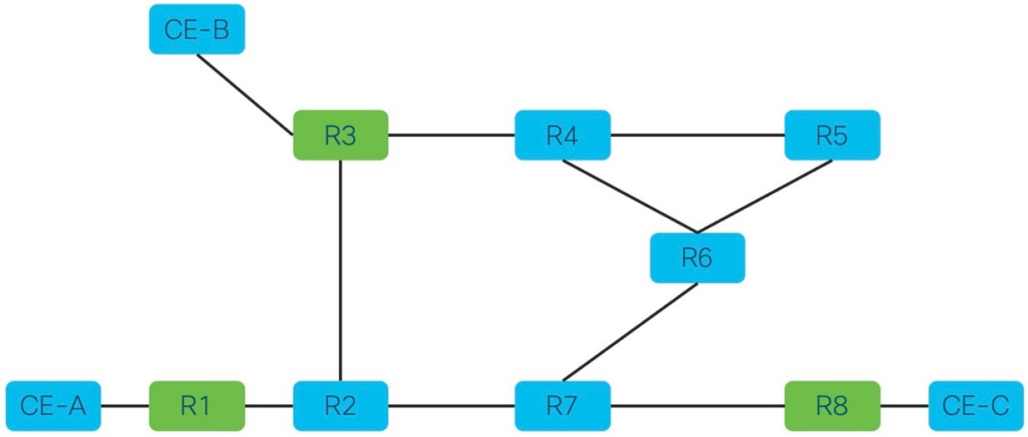
| Feature Name | Release Information | Feature Description |
|---|---|---|
| SRv6-Services: EVPN ELAN Layer 2 Gateway With Automated Steering To Flexible Algorithm Paths | Release 7.5.2 | This feature builds upon EVPN BGP signaling to provide Emulated Local Area Network (ELAN) multipoint-to-multipoint Ethernet services over an SRv6-based network.<br>You can enable automated steering of EVPN ELAN traffic into the path associated with a best-effort or Flex- Algorithm locator.<br>This feature combines the benefits of EVPN ELAN service and SRv6 Micro-SIDs.<br>For this feature, the **segment-routing srv6** option was added to the **evi** command:<br>evi (bridge-domain) |

You can transport EVPN ELAN bridged unicast and broadcast, unknown unicast, and multicast (BUM) traffic over an SRv6 network in the Micro-SID format. Relevant SRv6 headend and endpoint definitions are noted below:

- **H.Encaps.L2.Red**: This headend router operation involves reduced encapsulation of Layer 2 or Ether frames using an SRv6 Policy.
- **uDT2U**: This endpoint router operation involves traffic decapsulation and unicast MAC L2 table lookup. This is used for the EVPN bridging unicast traffic use case.
- **uDT2M**: This endpoint router operation involves traffic decapsulation and L2 table flooding. This is used for the EVPN bridging BUM traffic with ESI filtering use case.

**Note** For more information on SRv6 headend and endpoint behaviors, refer to Segment Routing over IPv6 Overview.

The following topology is used to explain this feature.




Topology pointers:

- Customer edge (CE) devices send traffic between each other over the SRv6 network. The CE devices are CE-A, CE-B, and CE-C.
- The SRv6 network devices transport customer traffic, and they are R1, R2 .. till R8.
- The provider edge (PE) devices, R1, R3 and R8, are displayed in green. The SRv6 EVPN configurations must be enabled on the PE devices since they participate in the EVPN EVI.

This is a high-level overview of the traffic flow from CE-A to CE-C:

1. CE-A sets the source and destination addresses of the L2 frame and sends it to the connected PE device, R1.
2. R1 looks up the destination MAC address in the frame. Based on its forwarding table, R1 performs an H.Encaps.L2.Red operation and adds the destination DT2U SRv6 SID (say, fccc:ccc1:a1:e000::) to the packet.
3. From R1, traffic is sent over the SRv6 network to destination PE device R8.
4. When R8 receives the traffic, it performs the uDT2U function - It decapsulates the packet, performs a destination MAC address lookup in its forwarding table, and sends the frame through the local interface to CE-C.

### Guidelines and Limitations

- For transporting BUM traffic, the BGP Route Reflector device should have an IOS XR release version 7.5.2 or later.

### Configure SRv6 EVPN Bridging

Enable the following configurations on the PE routers R1, R3 and R8 since they participate in the EVPN EVI.

**Note** Complete the steps in Segment Routing over IPv6 Overview before performing these steps.

### Associate SRv6 with EVPN

```
Router# configure terminal
Router(config)# evpn
```

Enable SRv6 under the EVPN mode and associate a global locator (**sample**, in this case) with EVPN.

```
Router(config-evpn)# segment-routing srv6
Router(config-evpn-srv6)# locator sample
Router(config-evpn-srv6)# exit
```

Associate an EVI-specific locator (**sample_evi_loc**) with EVI 1.

```
Router(config-evpn)# evi 1 segment-routing srv6
Router(config-evpn-instance)# locator sample_evi_loc
Router(config-evpn-instance)# commit
```

### Associate SRv6 with L2VPN

Associate the sub-interface to the bridge domain:

```
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd1
Router(config-l2vpn-bg-bd)# interface Hu0/0/0/0.1
Router(config-l2vpn-bg-bd-ac)# exit
```

Enable the **evi 1 segment-routing srv6** command under L2VPN bridge domain **bd1**.

```
Router(config-l2vpn-bg-bd)# evi 1 segment-routing srv6
Router(config-l2vpn-bg-bd-evi-srv6)# commit
```

### Verification

In this sample output, SRv6 EVPN ELAN traffic unicast and multicast SID information is displayed.

```
Router# show evpn evi vpn-id 1 detail

VPN-ID     Encap      Bridge Domain                 Type
---------- ---------- ----------------------------- ------------------
1          SRv6       bd1                           EVPN

..
   Stitching: Regular
   Unicast SID:  fccc:ccc1:a1:e000::
   Multicast SID: fccc:ccc1:a1:e001::
..
```

In this sample output, EVI 1 details, including the corresponding SID and EVPN MAC address details are displayed.

```
Router# show evpn evi vpn-id 1 mac

VPN-ID    Encap   MAC            IP        Nexthop      Label    SID
                  address        address
-----     ------ --------       --------   -------      -----         ---

1         SRv6    0010.3000.01d0  ::       Hu0/0/0/0.1    0     fccc:ccc1:a1:e000::
```

In this sample output, for the specified EVI and EVPN MAC address, SRv6 EVPN ELAN traffic details are displayed.

```
Router# show evpn evi vpn-id 1 mac 0010.3000.01d0 detail

VPN-ID Encap       MAC            IP       Nexthop      Label         SID
```

```
                  address         address
------ -----      ------          ------  --------       -----          ---
1     SRv6   ee03.0500.0130  ::     192.168.0.3  IMP-NULL fccc:ccc1:a3:e000::

   Ethernet Tag                           : 0
   Multi-paths Resolved                   : True
   Multi-paths Internal label             : None
   Local Static                           : No
   Remote Static                          : Yes
   Local Ethernet Segment                 : N/A
   Remote Ethernet Segment                : 0100.0205.acce.5500.0500
   Local Sequence Number                  : N/A
   Remote Sequence Number                 : 0
   Local Encapsulation                    : N/A
   Remote Encapsulation                   : SRv6
   Local E-Tree                           : Root
   Remote E-Tree                          : Root
   Remote matching E-Tree RT              : No
   Local AC-ID                            : 0x0
   Remote AC-ID                           : 0x13
```

In this sample output, for the specified EVI, multicast SID details are displayed.

```
Router# show evpn evi vpn-id 1 inclusive-multicast detail

VPN-ID        Encap           EtherTag             Originating IP
---           ----            -------              -------------
1             SRv6       0                    192.168.0.1
..
          TEPid: 0xffffffff
  PMSI Type: 6
  Nexthop: ::
  SR-TE Info: N/A
     SID: fccc:ccc1:a1:e001::
  Source: Local
  E-Tree: Root
..
```

In this sample output, for the specified MAC address, bridge domain information is displayed.

```
Router# show l2route evpn mac all | i ee03.0500.0130

Topo ID   Mac Address             Producer     Next Hop(s)
------    -----------             -------      -------------
1         ee03.0500.0130    L2VPN        ::ffff:10.0.0.10/IID/V6, N/A
```

In this sample output, SRv6 network locator and corresponding SID information are displayed.

**uDT2U** and **uDT2M** refer to SRv6 network endpoint operations. **uDT2U** indicates SRv6 traffic decapsulation, wherein EVPN bridged unicast traffic is forwarded out of the SR network. **uDT2M** indicates SRv6 traffic decapsulation, wherein EVPN bridged multicast traffic is forwarded out of the SR network.

```
Router# show segment-routing srv6 sid

*** Locator: 'sample_evi_loc' ***

SID                     Behavior      Context        Owner        State   RW
---                     --------      -------        -----        -----   --
fccc:ccc1:a1::          uN(PSP/USD)  'default':161   sidmgr        InUse   Y
fccc:ccc1:a1:e000::     uDT2U   7:0            l2vpn_srv6    InUse   Y
fccc:ccc1:a1:e001::     uDT2M   7:0            l2vpn_srv6    InUse   Y
```

In this sample output, CEF information is displayed, including SRv6 network endpoint details. **uDT2U** is an SRv6 network endpoint operation wherein SRv6 traffic is decapsulated and EVPN bridged unicast traffic is forwarded out of the SR network.

```
Router# show cef ipv6 fccc:ccc1:a1:e000:: detail

fccc:ccc1:a1:e000::, version 14, SRv6 Endpoint uDT2U, internal 0x1000001
0x0 (ptr 0x8ba26050) [1], 0x400 (0x8bbf7b58), 0x0 (0x92396138)

 Prefix Len 64, traffic index 0, precedence n/a, priority 0
  gateway array (0x8ba33e90) reference count 4, flags 0x0, source rib (7), 0 backups
                [5 type 3 flags 0x8401 (0x8baf8ca8) ext 0x0 (0x0)]

  LW-LDI[type=3, refc=1, ptr=0x8bbf7b58, sh-ldi=0x8baf8ca8]
  gateway array update type-time 1 Sep  8 11:46:51.242

 LDI Update time Sep  8 11:46:51.303
 LW-LDI-TS Sep  8 11:46:51.380
   via ::/128, 0 dependencies, weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0x8afdf120 0x0]
    next hop ::/128
    XConnect ID: 0x80000003
    Bridge ID: 0x1
    Shg ID: 0x1

    Load distribution: 0 (refcount 5)

    Hash  OK  Interface                 Address
    0     Y   recursive                 Lookup in table
```

In the following examples, SRv6 EVPN ELAN traffic-related IID information is displayed.

```
Router# show evpn internal-id vpn-id 3001 detail

VPN-ID  Encap  Ethernet Segment Id         EtherTag     Internal ID
-----   ------ -------------------------- ----------   -----------
1       SRv6   0001.0001.0001.1501.0015    0          ::ffff:10.0.0.4

   Summary pathlist:

 0x05000002 (P) 192.168.0.3 fccc:ccc1:a3:e000::
 0x05000002 (P) 192.168.0.3 fccc:ccc1:a4:e000::
```

```
Router# show cef vrf **iid ipv6 ::ffff:10.0.0.4

::ffff:10.0.0.4/128, version 39, SRv6 Headend, IID (EVPN-MH), internal 0x1000001 0x0 (ptr
0x8ba21798) [3], 0x0 (0x0), 0x0 (0x923967b0)
 Updated Sep  8 18:01:06.495
 Prefix Len 128, traffic index 0, precedence n/a, priority 0
  gateway array (0x8ba36018) reference count 1, flags 0x2010, source rib (7), 0 backups
                [1 type 3 flags 0x48441 (0x8baf9a28) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Sep  8 18:01:06.495
 LDI Update time Sep  8 18:01:06.495

  Level 1 - Load distribution: 0
  [0] via fccc:ccc1:a3::/128, recursive

   via fccc:ccc1:a3:e000::/128, 10 dependencies, recursive [flags 0x0]
    path-idx 0 NHID 0x0 [0x8ba24e78 0x0]
    next hop VRF - 'default', table - 0xe0800000
    next hop fccc:ccc1:a3::/128 via fccc:ccc1:a3::/48
    SRv6 H.Encaps.L2.Red SID-list { fccc:ccc1:a3:e000::}

    Load distribution: 0 (refcount 1)

    Hash  OK  Interface        Address
    0     Y   Hu0/0/0/0        remote
```

```
   via fccc:ccc1:a4::/128, 10 dependencies, recursive [flags 0x100]
    path-idx 0 NHID 0x0 [0x8ba24e78 0x0]
    next hop VRF - 'default', table - 0xe0800000
    next hop fccc:ccc1:a4::/128 via fccc:ccc1:a4::/48
    SRv6 H.Encaps.L2.Red SID-list {fccc:ccc1:a4:e000::}
```

In this sample output, SRv6 EVPN ELAN traffic-related IID information is displayed.

```
Router# show rib ipv6 iid

IID         Prefix                Context             Owner           State      RW
-------     ------                -------             -------         -----      --
0xa000001 ::ffff:10.0.0.1
[EVPN-ELAN:evi=7:esi=8300.fccc.ccc1.00a4.0000:nh=fccc:ccc1:a4:::eth_tag=0:type=0:encap=255:opaque=0]

                                                        l2vpn_iid        InUse      Y

0xa000002 ::ffff:10.0.0.2
[EVPN-ELAN:evi=8:esi=8300.fccc.ccc1.00a4.0000:nh=fccc:ccc1:a4:::eth_tag=0:type=0:encap=255:opaque=0]

                                                        l2vpn_iid        InUse      Y

0xa000003 ::ffff:10.0.0.3
[EVPN-ELAN:evi=9:esi=8300.fccc.ccc1.00a4.0000:nh=fccc:ccc1:a4:::eth_tag=0:type=0:encap=255:opaque=0]

                                                        l2vpn_iid        InUse      Y
```

# SRv6/MPLS L3 Service Interworking Gateway

SRv6/MPLS L3 Service Interworking Gateway enables you to extend L3 services between MPLS and SRv6 domains by providing service continuity on the control plane and data plane.

This feature allows for SRv6 L3VPN domains to interwork with existing MPLS L3VPN domains. The feature also allows a way to migrate from MPLS L3VPN to SRv6 L3VPN.

The SRv6/MPLS L3 Service Interworking Gateway provides both transport and service termination at the gateway node. The gateway generates both SRv6 VPN SIDs and MPLS VPN labels for all prefixes under the VRF configured for re-origination. The gateway supports traffic forwarding from MPLS domain to SRv6 domain by popping the MPLS VPN label, looking up the destination prefix, and pushing the appropriate SRv6 encapsulation. From SRv6 domain to MPLS domain, the gateway removes the outer IPv6 header, looks up the destination prefix, and pushes the VPN and next-hop MPLS labels.

VRFs on the gateway node are configured with 2 sets of route targets (RTs):

- MPLS L3VPN RTs
- SRv6 L3VPN RTs (called *stitching RTs*)

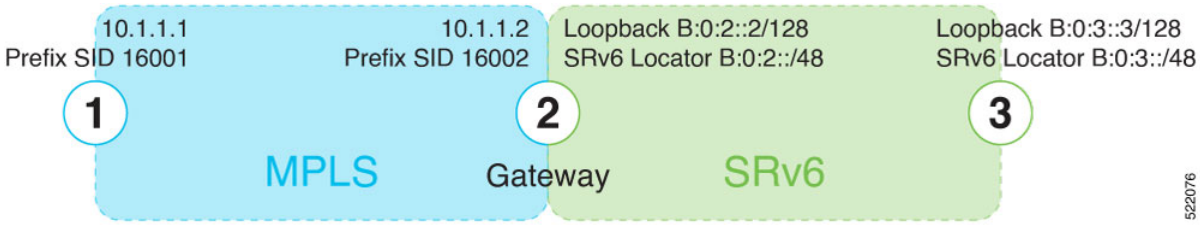
The gateway performs the following actions:

- Imports service routes received from one domain (MPLS or SRv6)
- Re-advertises exported service routes to the other domain (next-hop-self)
- Stitches the service on the data plane (uDT4/H.Encaps.Red ↔ service label)

### SRv6/MPLS L3 Service Interworking Gateway Scenarios

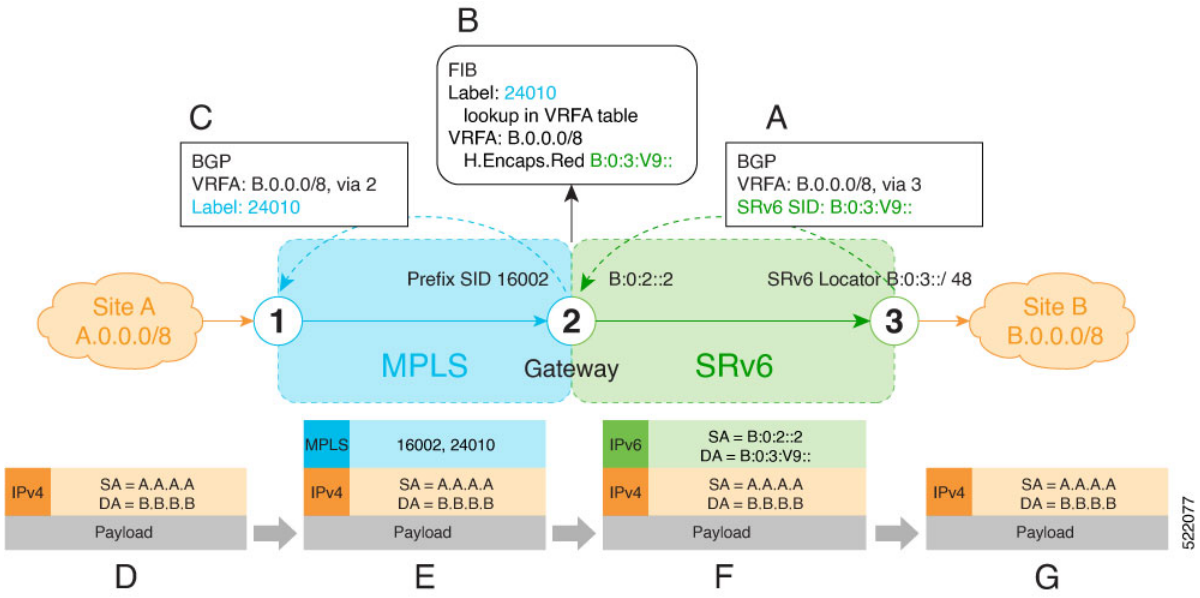
The following scenario is used to describe the gateway functionality:

- Node 1 is an L3VPN PE in the MPLS domain with an SR prefix SID label of 16001 for its Loopback interface 10.1.1.1/32.
- Node 2 is the SRv6/MPLS L3 Service Interworking Gateway. In the MPLS domain, it has an SR prefix SID label of 16002 for its Loopback interface 10.1.1.2/32. In the SRv6 domain, it has an SRv6 locator of B:0:2::/48 and Loopback interface B:0:2::2/128.
- Node 3 is an L3VPN PE in the SRv6 domain with SRv6 locator of B:0:3::/48 and Loopback interface B:0:3::3/128.




### Scenario 1: SRv6-to-MPLS Control-Plane Direction/MPLS-to-SRv6 Data-Plane Direction

The figure below describes the associated control-plane behaviors in the SRv6-to-MPLS direction for traffic in the MPLS-to-SRv6 data-plane direction.




**A**. Node 3 advertises a BGP L3VPN update for prefix B.0.0.0/8 with RD corresponding to VRFA, including the SRv6 VPN SID (B:0:3:V9::) assigned to this VRF, in the SRv6 domain.

**Note** SRv6 uDT4 function value "V9" is not a valid hex number, however it is used for illustration purposes to remind you of its connection to a VRF.

**B**. Node 2 (gateway) imports the BGP L3VPN update and programs its FIB:

- MPLS label 24010 is allocated for VRFA
- Prefix B.0.0.0/8 is programmed with an "SR Headend Behavior with Reduced Encapsulation in an SR Policy" function (H.Encaps.Red) of B:0:3:V9::

**Note** The gateway follows per-VRF label and per-VRF SID allocation methods.

**C**. Node 2 re-originates a BGP L3VPN update for the same prefix, including the MPLS VPN label (24010) allocated for the VRF, in the MPLS domain.

**D**. Site A sends traffic to an IPv4 prefix (B.B.B.B) of Site B

**E**. Node 1 encapsulates incoming traffic with the MPLS VPN label (24010) and the prefix SID MPLS label (16002) of the BGP next-hop (Node 2).

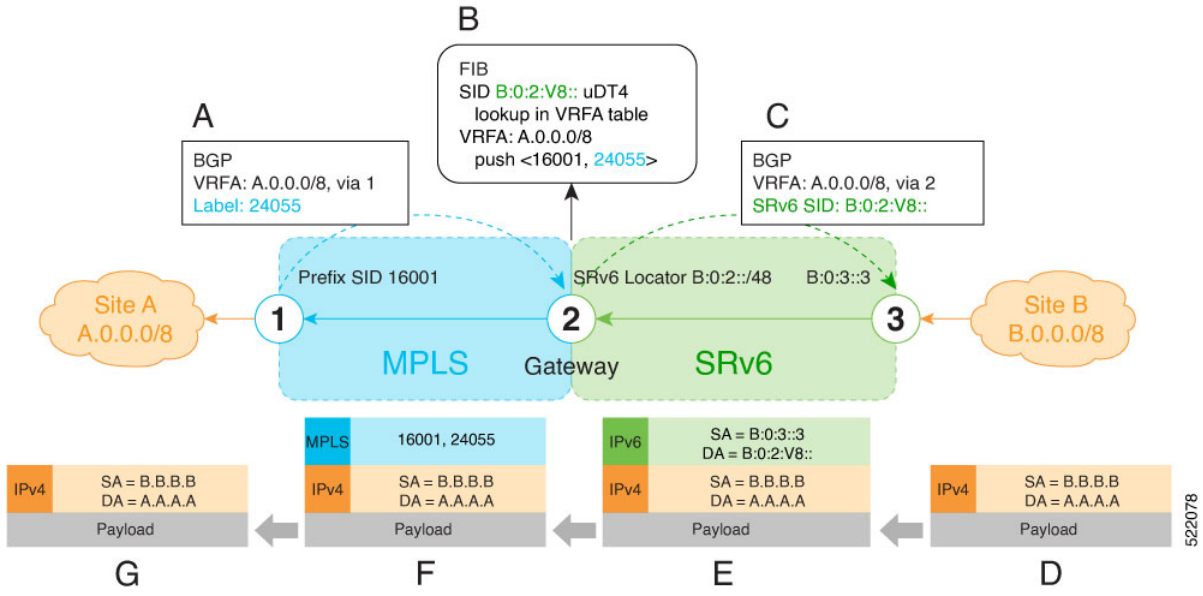
**F**. Node 2 performs the following actions:

- Pops the MPLS VPN label and looks up the destination prefix
- Encapsulates the payload in an outer IPv6 header with destination address (DA) equal to the H.Encaps.Red function (B:0:3:V9::)

**G**. Node 3 removes the outer IPv6 header, looks up the payload destination address (B.B.B.B), and forwards to Site B.

### Scenario 2: MPLS-to-SRv6 Control-Plane Direction/SRv6-to-MPLS Data-Plane Direction

The figure below describes the associated control-plane behaviors in the MPLS-to-SRv6 direction for traffic in the SRv6-to-MPLS data-plane direction.




**A**. Node 1 advertises a BGP L3VPN update for prefix A.0.0.0/8 with RD corresponding to VRFA, including the MPLS VPN label (24055) assigned to this VRF, in the MPLS domain.

**B**. Node 2 (gateway) imports the BGP L3VPN update and programs its FIB:

- Prefix A.0.0.0/8 is programmed to impose an MPLS VPN label (24055) and the prefix SID MPLS label (16001) of the BGP next-hop (Node 1)
- "Endpoint with decapsulation and IPv4 table lookup" function (uDT4) of B:0:2:V8:: is allocated to VRFA

**Note** SRv6 uDT4 function value "V8" is not a valid hex number, however it is used for illustration purposes to remind you of its connection to a VRF.

**Note** The gateway follows per-VRF label and per-VRF SID allocation methods.

**C**. Node 2 re-originates a BGP L3VPN update for the same prefix, including the uDT4 function (B:0:2:V8::) allocated for the VRF, in the SRv6 domain.

**D**. Site B sends traffic to an IPv4 prefix (A.A.A.A) of Site A.

**E**. Node 3 Encapsulates the payload in an outer IPv6 header with destination address (DA) equal to the uDT4 function (B:0:2:V8::).

**F**. Node 2 performs the following actions:

- Removes the outer IPv6 header and looks up the destination prefix
- Pushes the MPLS VPN label (24055) and the prefix SID MPLS label (16001) of the BGP next-hop (Node 1)

**G**. Node 1 pops the MPLS VPN label, looks up the payload destination address (A.A.A.A), and forwards to Site A.

### Example

Leveraging the topology described in the above use-case, this example shows the SRv6/MPLS L3 Service Interworking Gateway configuration required at Node 2.

The following configuration shows how to enable SRv6 with locator and configure encapsulation parameters:

```
segment-routing
 srv6
  encapsulation
   source-address B:0:2::2
  !
  locators
   locator LOC1
    prefix B:0:2::/48
   !
  !
 !
!
```

The following configuration shows how to configure a VPNv4 VRF with the following route targets (RTs):

- 1111:1, RT used for MPLS L3VPN
- 2222:1, RT used for SRv6 L3VPN (stitching RT)

```
vrf ACME
 address-family ipv4 unicast
  import route-target
   1111:1
   2222:1 stitching
  !
  export route-target
   1111:1
   22222:1 stitching
  !
 !
!
```

The following configuration shows how to configure SRv6/SRv6 VPNs under BGP:

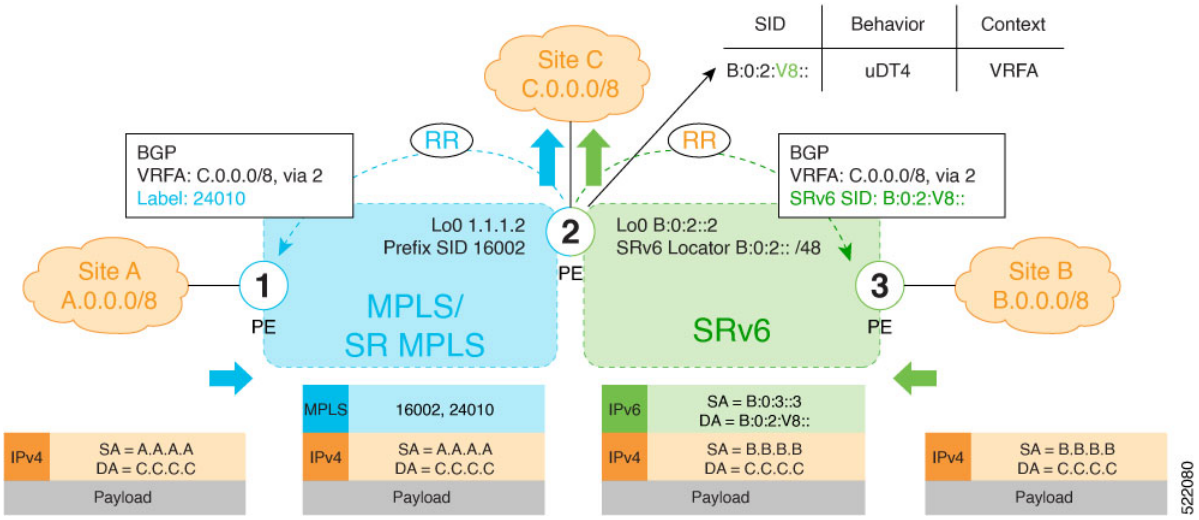
```
router bgp 100
 segment-routing srv6
  locator LOC1
 !
 neighbor 10.1.1.1
  address-family vpnv4 unicast
   import re-originate stitching-rt
   route-reflector-client
   advertise vpnv4 unicast re-originated
 !
 neighbor B:0:3::1
  address-family vpnv4 unicast
   import stitching-rt re-originate
   route-reflector-client
   encapsulation-type srv6
   advertise vpnv4 unicast re-originated stitching-rt
 !
 vrf ACME
  address-family ipv4 unicast
   enable label-mode
   segment-routing srv6
```

# SRv6/MPLS Dual-Connected PE

A PE router can support IPv4 L3VPN service for a given VRF with both MPLS and SRv6. This is MPLS and SRv6 L3VPNv4 co-existence scenario and is sometimes referred to as dual-connected PE.

In the figure below, node 2 is a dual-connected PE to Site C, providing:

- MPLS/IPv4 L3VPN between Site A and Site C
- SRv6/IPv4 L3VPN between Site B and Site C




### Configure BGP to Support Dual-Mode

#### Enable MPLS Label Allocation

Use the **router bgp** *as-number* **vrf** *WORD* **address-family ipv4 unicast mpls alloc enable** command under the VRF address-family to enable per-prefix mode for MPLS labels. Additionally, use the **router bgp** *as-number* **vrf** *WORD* **address-family ipv4 unicast label mode** {**per-ce** | **per-vrf**} command to choose the type of label allocation.

```
Router(config)# router bgp 100
Router(config-bgp)# vrf blue
Router(config-bgp-vrf)# rd 1:10
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# mpls alloc enable
Router(config-bgp-vrf-af)# label mode per-ce
Router(config-bgp-vrf-af)# segment-routing srv6
Router(config-bgp-vrf-af-srv6)# alloc mode per-ce
Router(config-bgp-vrf-af-srv6)# exit
Router(config-bgp-vrf-af)# exit
Router(config-bgp-vrf)# exit
Router(config-bgp)#
```

#### Configure Encaps on Neighbor to Send the SRv6 SID Toward the SRv6 Dataplane

By default, if a VRF prefix has both an MPLS label and an SRv6 SID, the MPLS label is sent when advertising the prefix to the PE. To advertise a VRF prefix with an SRv6 SID to an SRv6 session, use the **encapsulation-type srv6** command under the neighbor VPN address-family.

```
Router(config-bgp)# neighbor 192::6
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# encapsulation-type srv6
Router(config-bgp-nbr-af)# exit
```

#### Running Config

```
router bgp 100
 neighbor 192::6
  remote-as 1
  address-family ipv4 unicast
   encapsulation-type srv6
```

```
  !
 !
 vrf blue
  rd 1:10
  address-family ipv4 unicast
   mpls alloc enable
   label mode per-ce
   segment-routing srv6
    alloc mode per-ce
   !
  !
 !
!
```

# SRv6 SID Information in BGP-LS Reporting

BGP Link-State (BGP-LS) is used to report the topology of the domain using nodes, links, and prefixes. This feature adds the capability to report SRv6 Segment Identifier (SID) Network Layer Reachability Information (NLRI).

The following NLRI has been added to the BGP-LS protocol to support SRv6:

- Node NLRI: SRv6 Capabilities, SRv6 MSD types
- Link NLRI: End.X, LAN End.X, and SRv6 MSD types
- Prefix NLRI: SRv6 Locator
- SRv6 SID NLRI (for SIDs associated with the node): Endpoint Function, BGP-EPE Peer Node/Set

This example shows how to distribute IS-IS SRv6 link-state data using BGP-LS:

```
Router(config)# router isis 200
Router(config-isis)# distribute link-state instance-id 200
```

**Note** It is still possible to ping or trace a SID:

- **ping** B:k:F::
- **traceroute** B:k:F::

It is possible to use a list of packed carriers to ping or trace a SID, to ping or trace route, use **<destination SID> via srv6-carriers <list of packed carriers>**

# DHCPv4 Relay Agent and Proxy Support over SRv6

This feature introduces support for DHCPv4 Relay Agent and Proxy over SRv6.

An IOS XR router can act as a DHCPv4 relay agent/proxy with a DHCPv4 server connected over an SRv6 network.

The following functionality is supported:

- DHCPv4 relay agent/proxy over SRv6 with DHCPv4 server (helper-address) located in default VRF (global)
- DHCPv4 relay agent/proxy over SRv6 with DHCPv4 server (helper-address) located in non-default VRF
- DHCPv4 relay agent/proxy on interfaces associated with a default VRF (global)
- DHCPv4 relay agent/proxy on interfaces associated with a non-default VRF
- DHCPv4 relay agent/proxy on Ethernet physical interfaces
- DHCPv4 relay agent/proxy on Ethernet bundle interfaces

For information on configuring DHCPv4 relay agent and proxy, refer to the "Implementing the Dynamic Host Configuration Protocol" chapter in the *IP Addresses and Services Configuration Guide for Cisco NCS540 Series Routers* .

# DHCPv6 Relay Agent Support over SRv6

***Table 10: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| DHCPv6 Relay Agent Support on SRv6 | Release 7.2.2 | An IOS XR router can act as a DHCPv6 relay agent with a DHCPv6 server connected over an SRv6 network.<br>A DHCP relay agent is a host that forwards DHCP packets between clients and servers that do not reside on a shared physical subnet. |

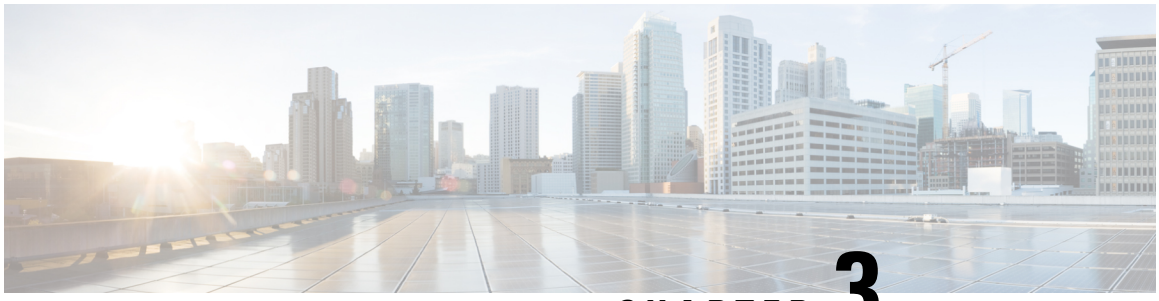
This feature introduces support for DHCPv6 Relay Agent over SRv6.

An IOS XR router can act as a DHCPv6 relay agent with a DHCPv6 server connected over an SRv6 network.

The following functionality is supported:

- DHCPv6 relay agent over SRv6 with DHCPv6 server (helper-address) located in default VRF (global)
- DHCPv6 relay agent over SRv6 with DHCPv6 server (helper-address) located in non-default VRF
- DHCPv6 relay agent on interfaces associated with a default VRF (global)
- DHCPv6 relay agent on interfaces associated with a non-default VRF
- DHCPv6 relay agent on Ethernet physical interfaces
- DHCPv6 relay agent on Ethernet bundle interfaces

For information on configuring DHCPv6 relay agent, refer to the "Implementing the Dynamic Host Configuration Protocol" chapter in the *IP Addresses and Services Configuration Guide for Cisco NCS540 Series Routers*.



CHAPTER 3

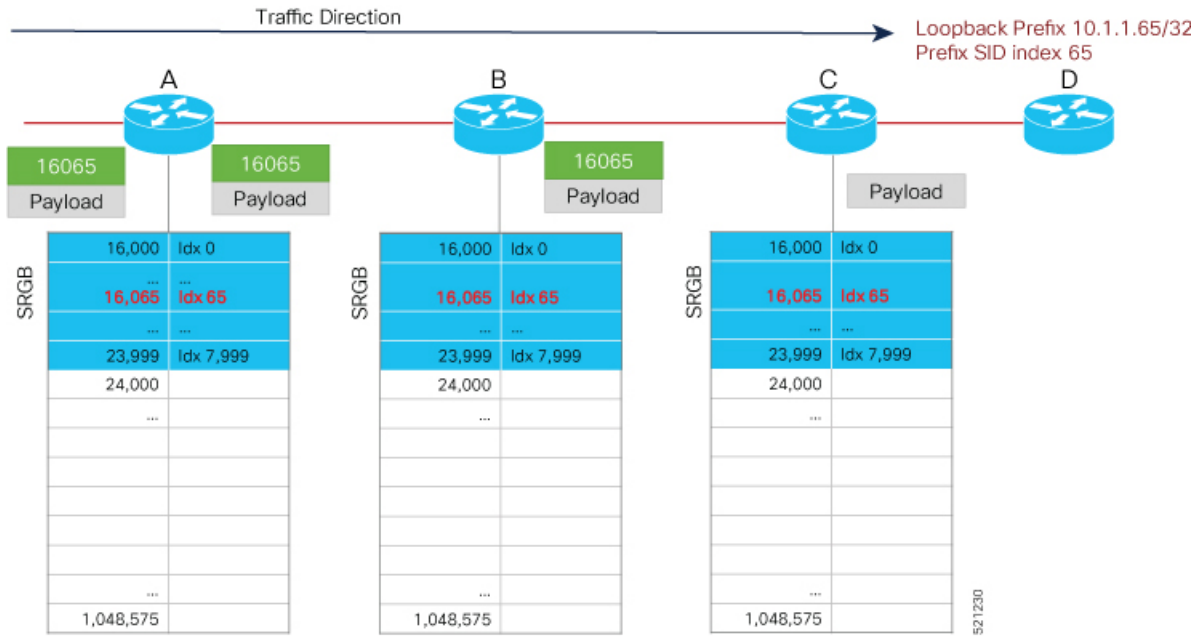
# Configure Segment Routing Global Block and Segment Routing Local Block

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



## About the Segment Routing Global Block

The Segment Routing Global Block (SRGB) is a range of labels reserved for Segment Routing global segments. A prefix-SID is advertised as a domain-wide unique index. The prefix-SID index points to a unique label within the SRGB range. The index is zero-based, meaning that the first index is 0. The MPLS label assigned to a prefix is derived from the Prefix-SID index plus the SRGB base. For example, considering an SRGB range of 16,000 to 23,999, a prefix 10.1.1.65/32 with prefix-SID index of **65** is assigned the label value of **16065**.




To keep the configuration simple and straightforward, we strongly recommended that you use a homogenous SRGB (meaning, the same SRGB range across all nodes). Using a heterogenous SRGB (meaning, a different SRGB range of the same size across nodes) is also supported but is not recommended.

### Behaviors and Limitations

- The default SRGB in IOS XR has a size of 8000 starting from label value 16000. The default range is 16000 to 23,999. With this size, and assuming one loopback prefix per router, an operator can assign prefix SIDs to a network with 8000 routers.
- There are instances when you might need to define a different SRGB range. For example:
  - Non-IOS XR nodes with a SRGB range that is different than the default IOS XR SRGB range.
  - The default SRGB range is not large enough to accommodate all required prefix SIDs.
- A non-default SRGB can be configured following these guidelines:
  - The SRGB starting value can be configured anywhere in the dynamic label range space (16,000 to 1,048,575).
  - In Cisco IOS XR release earlier than 6.6.3, the SRGB can have a maximum configurable size of 262,143.
  - In Cisco IOS XR release 6.6.3 and later, the SRGB can be configured to any size value that fits within the dynamic label range space.
- Allocating an SRGB label range does not mean that all the labels in this range are programmed in the forwarding table. The label range is just reserved for SR and not available for other purposes. Furthermore, a platform may limit the number of local labels that can be programmed.
- We recommend that the non-default SRGB be configured under the **segment-routing** global configuration mode. By default, all IGP instances and BGP use this SRGB.

- You can also configure a non-default SRGB under the IGP, but it is not recommended.

### SRGB Label Conflicts

When you define a non-default SRGB range, there might be a label conflict (for example, if labels are already allocated, statically or dynamically, in the new SRGB range). The following system log message indicates a label conflict:

```
%ROUTING-ISIS-4-SRGB_ALLOC_FAIL : SRGB allocation failed: 'SRGB reservation not
successful for [16000,80000], SRGB (16000 80000, SRGB_ALLOC_CONFIG_PENDING, 0x2)
(So far 16 attempts). Make sure label range is free'
```

To remove this conflict, you must reload the router to release the currently allocated labels and to allocate the new SRGB.

After the system reloads, LSD does not accept any dynamic label allocation before IS-IS/OSPF/BGP have registered with LSD. Upon IS-IS/OSPF/BGP registration, LSD allocates the requested SRGB (either the default range or the customized range).

After IS-IS/OSPF/BGP have registered and their SRGB is allocated, LSD starts serving dynamic label requests from other clients.

**Note** To avoid a potential router reload due to label conflicts, and assuming that the default SRGB size is large enough, we recommend that you use the default IOS XR SRGB range.

**Note** Allocating a non-default SRGB in the upper part of the MPLS label space increases the chance that the labels are available and a reload can be avoided.

**Caution** Modifying a SRGB configuration is disruptive for traffic and may require a reboot if the new SRGB is not available entirely.

# About the Segment Routing Local Block

A local segment is automatically assigned an MPLS label from the dynamic label range. In most cases, such as TI-LFA backup paths and SR-TE explicit paths defined with IP addresses, this dynamic label allocation is sufficient. However, in some scenarios, it could be beneficial to allocate manually local segment label values to maintain label persistency. For example, an SR-TE policy with a manual binding SID that is performing traffic steering based on incoming label traffic with the binding SID.

The Segment Routing Local Block (SRLB) is a range of label values preserved for the manual allocation of local segments, such as adjacency segment identifiers (adj-SIDs) , Layer 2 adj-SIDs, binding SIDs (BSIDs), and BGP peering SIDs. These labels are locally significant and are only valid on the nodes that allocate the labels.

### Behaviors and Limitations

- The default SRLB has a size of 1000 starting from label value 15000; therefore, the default SRLB range goes from 15000 to 15,999.
- A non-default SRLB can be configured following these guidelines:
  - The SRLB starting value can be configured anywhere in the dynamic label range space (16,000 to 1,048,575).
  - In Cisco IOS XR release earlier than 6.6.3, the SRLB can have a maximum configurable size of 262,143.
  - In Cisco IOS XR release 6.6.3 and later, the SRLB can be configured to any size value that fits within the dynamic label range space.

### SRLB Label Conflicts

When you define a non-default 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 state). The previous SRLB range (active) will continue to be in use.

To remove this conflict, you must reload the router to release the currently allocated labels and to allocate the new SRLB.

**Caution** You can use the **clear segment-routing local-block discrepancy all** command to clear label conflicts. However, using this command is disruptive for traffic since it forces all other MPLS applications with conflicting labels to allocate new labels.

**Note** To avoid a potential router reload due to label conflicts, and assuming that the default SRGB size is large enough, we recommend that you use the default IOS XR SRLB range.

**Note** Allocating a non-default SRLB in the upper part of the MPLS label space increases the chance that the labels are available and a reload can be avoided.

# Setup a Non-Default Segment Routing Global Block Range

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:** | Enters mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# configure` | |
| **Step 2** | **segment-routing global-block** *starting_value ending_value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# segment-routing global-block 16000 80000` | 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. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Use the **show mpls label table** [**label** *label-value*] command to verify the SRGB configuration:

```
Router# show mpls label table label 16000 detail
Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
0     16000   ISIS(A):1                       InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=16000, size=64001)
```

### 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:** | Enters mode. |

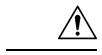
| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# configure` | |
| **Step 2** | **segment-routing local-block** *starting_value ending_value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# segment-routing local-block 30000 30999` | Enter the lowest value that you want the SRLB range to include as the starting value. Enter the highest value that you want the SRLB range to include as the ending value. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Use the **show mpls label table** [**label** *label-value*] [**detail**] command to verify the SRLB configuration:

```
Router# show mpls label table label 30000 detail

Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
0     30000   LSD(A)                          InUse  No
  (Lbl-blk SRLB, vers:0, (start_label=30000, size=1000, app_notify=0)

Router# show segment-routing local-block inconsistencies

No inconsistencies
```

The following example shows an SRLB label conflict in the range of 30000 and 30999. Note that the default SRLB is active and the configured SRLB is pending:

```
Router(config)# segment-routing local-block 30000 30999

%ROUTING-MPLS_LSD-3-ERR_SRLB_RANGE : SRLB allocation failed: 'SRLB reservation not successfull

for [30000,30999]. Use with caution 'clear segment-routing local-block discrepancy all'
command
to force srlb allocation'
```



**Caution** You can use the **clear segment-routing local-block discrepancy all** command to clear label conflicts. However, using this command is disruptive for traffic since it forces all other MPLS applications with conflicting labels to allocate new labels.

```
Router# show mpls label table label 30000 detail

Table Label   Owner                           State  Rewrite
----- ------- ------------------------------- ------ -------
0     30000   LSD(A)                          InUse  No
  (Lbl-blk SRLB, vers:0, (start_label=30000, size=1000, app_notify=0)

Router# show segment-routing local-block inconsistencies
SRLB inconsistencies range: Start/End: 30000/30999

Router# show mpls lsd private | i SRLB

SRLB Lbl Mgr:
   Current Active SRLB block      = [15000, 15999]
   Configured Pending SRLB block  = [30000, 30999]
```

Reload the router to release the currently allocated labels and to allocate the new SRLB:

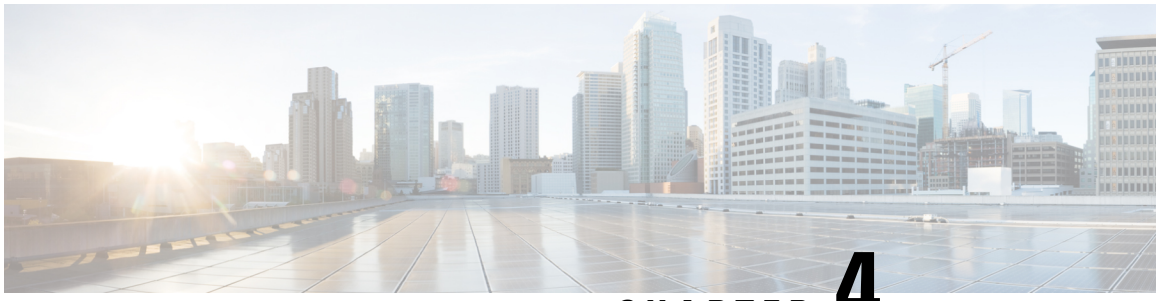
```
Router# reload

Proceed with reload? [confirm]yes
```

After the system is brought back up, verify that there are no label conflicts with the SRLB configuration:

```
Router# show mpls lsd private | i SRLB

SRLB Lbl Mgr:
   Current Active SRLB block      = [30000, 30999]
   Configured Pending SRLB block  = [0, 0]

Router# show segment-routing local-block inconsistencies

No inconsistencies
```

### What to do next

Configure adjacency SIDs and enable segment routing.



CHAPTER 4

# Configure Segment Routing for IS-IS Protocol

Integrated Intermediate System-to-Intermediate System (IS-IS), Internet Protocol Version 4 (IPv4), is a standards-based Interior Gateway Protocol (IGP). The Cisco IOS XR software implements the IP routing capabilities described in International Organization for Standardization (ISO)/International Engineering Consortium (IEC) 10589 and RFC 1995, and adds the standard extensions for single topology and multitopology IS-IS for IP Version 6 (IPv6).

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

**Note** For additional information on implementing IS-IS on your router , see the *Implementing IS-IS* module in the *Routing Configuration Guide for Cisco NCS 540 Series Routers*.



# 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
- 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.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis isp` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>**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** ]<br>**Example:**<br>`RP/0/RP0/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** }]<br>**Example:**<br>`RP/0/RP0/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** | **router-id loopback** *loopback interface used for prefix-sid*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-af)# router-id loopback0` | Configures router ID for each address-family (IPv4/IPv6).<br>IS-IS advertises the router ID in TLVs 134 (for IPv4 address family) and 140 (for IPv6 address family). Required when traffic engineering is used. |
| **Step 6** | **segment-routing mpls** [**sr-prefer**]<br>**Example:** | Segment routing is enabled by the following actions: |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-isis-af)# segment-routing mpls` | • MPLS forwarding is enabled on all interfaces where IS-IS is active.<br>• 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.<br>• The prefix-SIDs locally configured are advertised.<br>Use the **sr-prefer** keyword to set the preference of segment routing (SR) labels over label distribution protocol (LDP) labels. |
| **Step 7** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-af)# exit`<br>`RP/0/RP0/CPU0:router(config-isis)# exit` | |
| **Step 8** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

Configure the prefix SID.

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

A prefix segment identifier (SID) is associated with an IP prefix. The prefix SID is manually configured from the segment routing global block (SRGB) range of labels. A prefix SID is configured under the loopback

interface with the loopback address of the node as the prefix. The prefix segment steers the traffic along the shortest path to its destination.

A prefix SID can be a node SID or an Anycast SID. A node SID is a type of prefix SID that identifies a specific node. An Anycast SID is a type of prefix SID that identifies a set of nodes, and is configured with n-flag clear. The set of nodes (Anycast group) is configured to advertise a shared prefix address and prefix SID. Anycast routing enables the steering of traffic toward multiple advertising nodes. Packets addressed to an Anycast address are forwarded to the topologically nearest nodes.

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.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>• You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command. |
| **Step 3** | **interface Loopback** *instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis)# interface Loopback0` | Specifies the loopback interface and instance. |
| **Step 4** | **address-family** { **ipv4** \| **ipv6** } [ **unicast** ]<br>**Example:**<br>The following is an example for ipv4 address family:<br>`RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast` | Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **prefix-sid** [**algorithm** *algorithm-number*] {**index** *SID-index* \| **absolute** *SID-value*} [**n-flag-clear**] [**explicit-null** ]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if-af)# prefix-sid index 1001`<br>`RP/0/RP0/CPU0:router(config-isis-if-af)# prefix-sid absolute 17001` | Configures the prefix-SID index or absolute value for the interface.<br>Specify **algorithm** *algorithm-number* to configure SR Flexible Algorithm.<br>Specify **index** *SID-index* for each node to create a prefix SID based on the lower boundary of the SRGB + the index.<br>Specify **absolute** *SID-value* for each node to create a specific prefix SID within the SRGB.<br>By default, the n-flag is set on the prefix-SID, indicating that it is a node SID. For specific prefix-SID (for example, Anycast prefix-SID), enter the `n-flag-clear` keyword. IS-IS does not set the `N` flag in the prefix-SID sub Type Length Value (TLV).<br>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. |
| **Step 6** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Verify the prefix-SID configuration:

```
RP/0/RP0/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

<...>
  Metric: 0          IP-Extended 10.0.0.1/32
    Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0

<...>
```

# Weighted Anycast SID-Aware Path Computation

*Table 11: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Weighted Anycast SID-Aware Path Computation | Release 7.3.1 | This feature extends Anycast SIDs with weighted nodes.<br>Weighted Anycast nodes advertise a cost (weight) along with the Anycast SID. Traffic is then distributed according to the weights.<br>Weighted Anycast SIDs allow for highly available paths with node redundancy and path optimality that provide Fast ReRoute (FRR) for node failure of service provider edge (PE) routers and ABR/ASBRs nodes in multi-domain networks. |

The Weighted Anycast SID feature extends Anycast SIDs with weighted nodes.

Anycast routing enables the steering of traffic toward multiple advertising nodes, providing load-balancing and redundancy. Packets addressed to an Anycast address are forwarded to the topologically nearest nodes. With the default (unweighted) behavior, the traffic is load-balanced across each node in the group evenly.

Weighted Anycast nodes advertise a cost along with the Anycast SID. This cost serves as a weight. Traffic to the SID is then distributed according to the weights.

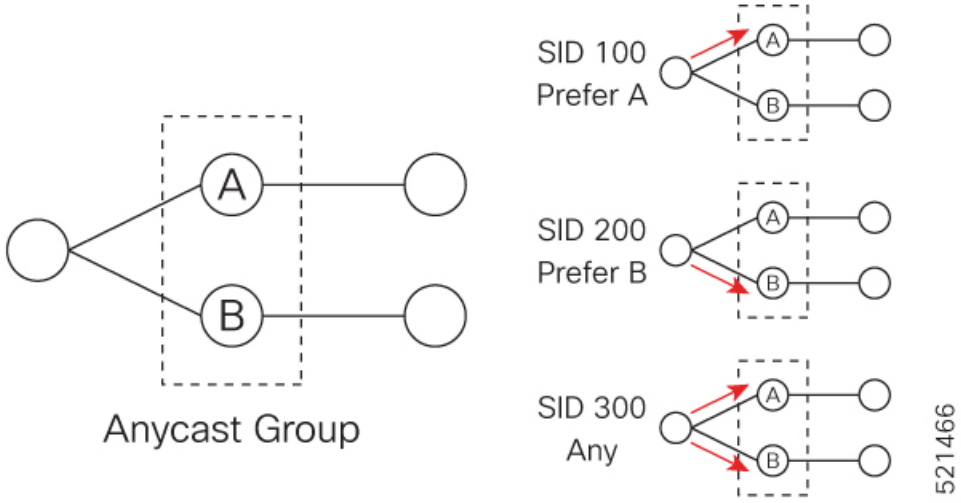
Weighted Anycast SIDs allow for highly available paths with node redundancy and path optimality that provide FRR for node failure of service provider edge (PE) routers and ABR/ASBR nodes in multi-domain networks.

In addition, Weighted Anycast SIDs allow for scaled computation at the PCE of multi-domain paths.

The native SR path computation algorithms are augmented to compute optimum paths relying on Weighted Anycast SIDs during path encoding.

Consider the example depicted below. Nodes A and B are part of the same Anycast groups, represented by different SIDs (100, 200, 300).

- SID 100 sends traffic preferentially to node A
- SID 200 sends traffic preferentially to node B
- SID 300 sends traffic equally to both nodes

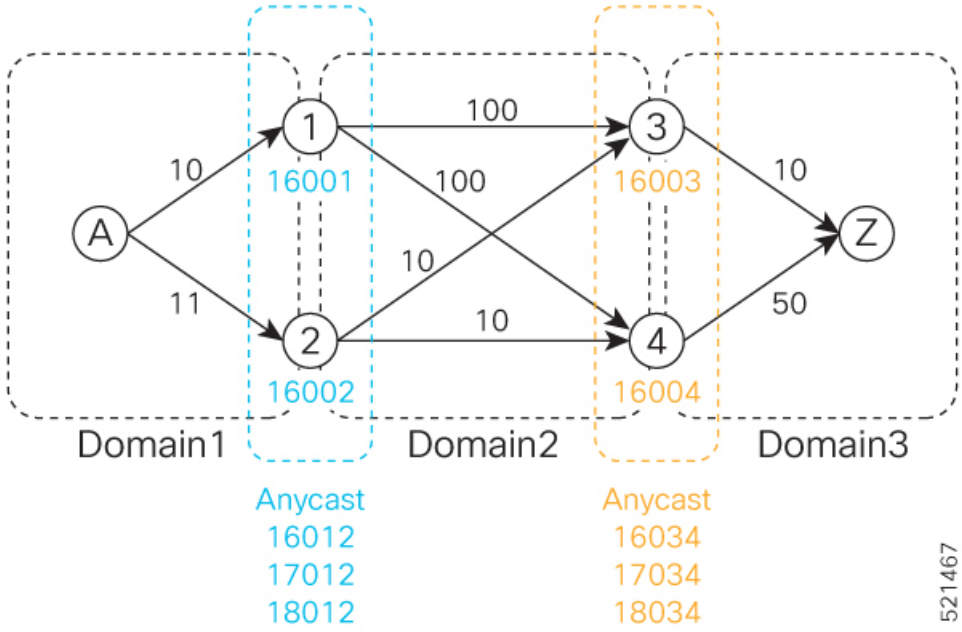



The Anycast replacement algorithm runs after an SR-TE path has been computed. It examines the prefix SIDs in the path and swaps them with Anycast SIDs that contain the same node. The new paths are checked against the original constraints and kept if suitable.

If a node is part of multiple Anycast groups, the algorithm considers them according to their weights.

### Example

The following figure shows 3 isolated IGP domains without redistribution and without BGP 3107. Each Area Border Router (ABR) 1 through 4 is configured with a node SID. The link delays are also shown.




ABRs 1 and 2 share the following Anycast SIDs:

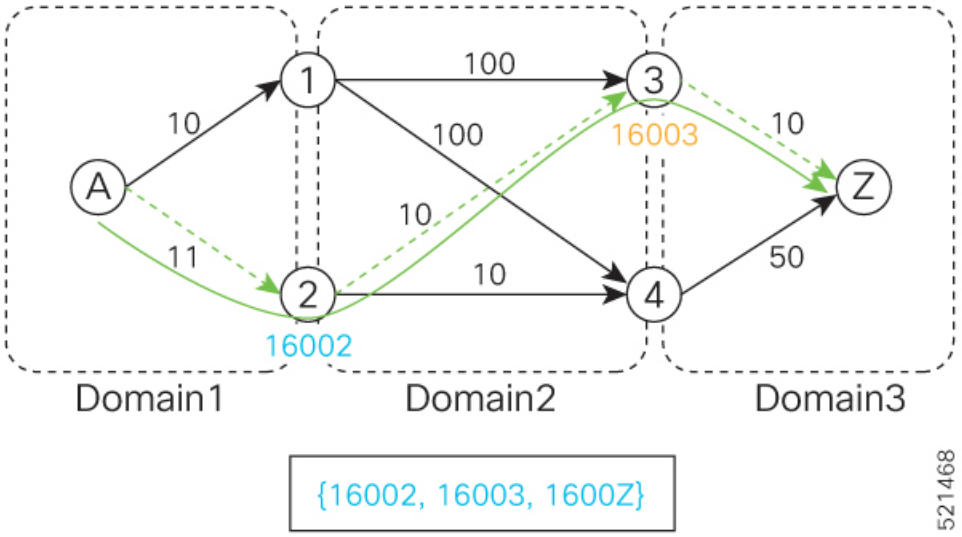
- 16012 – sends traffic to either Node 1 or 2 (the topologically nearest node)
- 17012 – sends traffic preferentially to Node 1
- 18012 – sends traffic preferentially to Node 2

ABRs 3 and 4 share the following Anycast SIDs:

- 16034 – sends traffic either Node 3 or 4 (the topologically nearest node)
- 17034 – sends traffic preferentially to Node 3
- 18034 – sends traffic preferentially to Node 4

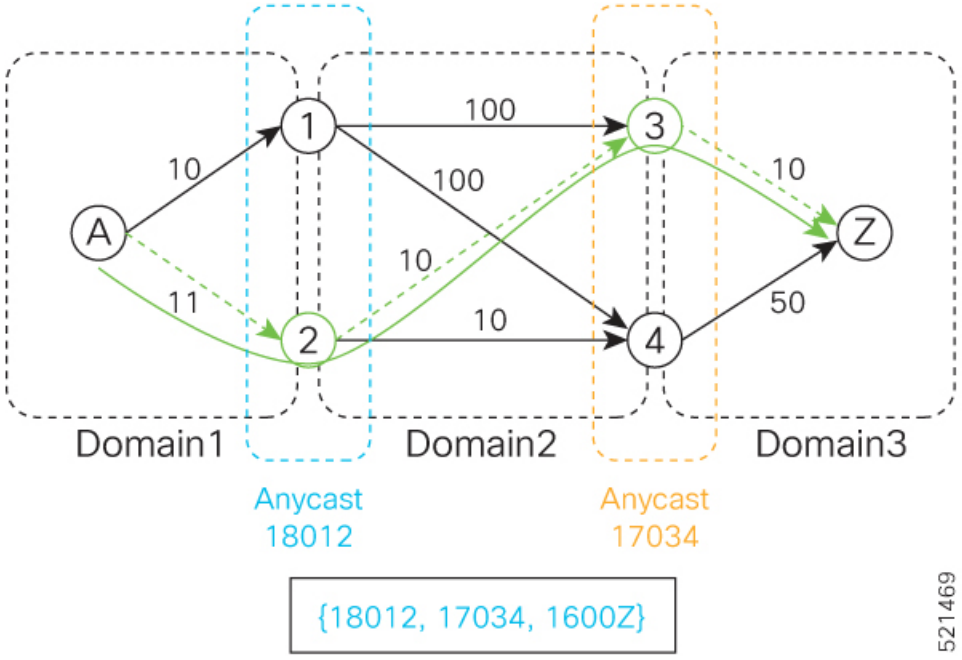
Consider the case where routers A and Z are provider edge (PE) routers in the same VPN. Router A receives a VPN route with BGP next-hop to router Z. Router A resolves the SR path to router Z using SR-ODN or SR-PCE.

Before considering Anycast SIDs, the head-end router or SR-PCE computes the SID list.




In this case, the optimized computed path from router A to router Z is 16002 > 16003 > 1600Z.

Using the weighted Anycast-encoded SID list, the optimized computed path from router A to router Z is 18012 > 17034 > 1600Z. This path has a cumulative delay of 31.




Using node SIDs, failures inside each domain (for example, links) benefit from fast TI-LFA convergence. However, failures of the ABR nodes would be dependent on SR-PCE reoptimization.

Using weighted Anycast SIDs, failures of the ABR nodes and failures inside each domain benefit from fast TI-LFA convergence.

### Configuration

Based on the topology in Figure *NN*, this example shows the Weighted Anycast SID configuration of ABRs 1 and 2.

### ABR 1 Configuration

```
RP/0/RSP0/CPU0:ios(config)# router isis 1
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback0
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 16001
RP/0/RSP0/CPU0:ios(config-isis-if-af)# exit
RP/0/RSP0/CPU0:ios(config-isis-if)# exit
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback1
RP/0/RSP0/CPU0:ios(config-isis-if)# prefix-attributes anycast
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 16012
RP/0/RSP0/CPU0:ios(config-isis-if-af)# exit
RP/0/RSP0/CPU0:ios(config-isis-if)# exit
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback2
RP/0/RSP0/CPU0:ios(config-isis-if)# prefix-attributes anycast
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# metric 1
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 17012
RP/0/RSP0/CPU0:ios(config-isis-if-af)# exit
RP/0/RSP0/CPU0:ios(config-isis-if)# exit
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback3
RP/0/RSP0/CPU0:ios(config-isis-if)# prefix-attributes anycast
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# metric 100000
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 18012
```

### Running Config

```
router isis 1
 interface Loopback0
  address-family ipv4 unicast
   prefix-sid absolute 16001 // Node SID
  !
 !
 interface Loopback1
  prefix-attributes anycast
  address-family ipv4 unicast
   prefix-sid absolute 16012 //Anycast SID - (prefer node 1 or 2)
  !
 !
 interface Loopback2
  prefix-attributes anycast
  address-family ipv4 unicast
   metric 1
   prefix-sid absolute 17012 // Weighted Anycast SID (prefer node 1)
  !
 !
 interface Loopback3
  prefix-attributes anycast
  address-family ipv4 unicast
   metric 100000
   prefix-sid absolute 18012 // Weighted Anycast SID (prefer node 2)
  !
 !
!
end
```

### ABR 2 Configuration

```
RP/0/RSP0/CPU0:ios(config)# router isis 1
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback0
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 16002
RP/0/RSP0/CPU0:ios(config-isis-if-af)# exit
RP/0/RSP0/CPU0:ios(config-isis-if)# exit
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback1
RP/0/RSP0/CPU0:ios(config-isis-if)# prefix-attributes anycast
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 16012
RP/0/RSP0/CPU0:ios(config-isis-if-af)# exit
RP/0/RSP0/CPU0:ios(config-isis-if)# exit
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback2
RP/0/RSP0/CPU0:ios(config-isis-if)# prefix-attributes anycast
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# metric 100000
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 17012
RP/0/RSP0/CPU0:ios(config-isis-if-af)# exit
RP/0/RSP0/CPU0:ios(config-isis-if)# exit
RP/0/RSP0/CPU0:ios(config-isis)# interface Loopback3
RP/0/RSP0/CPU0:ios(config-isis-if)# prefix-attributes anycast
RP/0/RSP0/CPU0:ios(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:ios(config-isis-if-af)# metric 1
RP/0/RSP0/CPU0:ios(config-isis-if-af)# prefix-sid absolute 18012
```

### Running Config

```
router isis 1
 interface Loopback0
  address-family ipv4 unicast
   prefix-sid absolute 16002 // Node SID
  !
 !
 interface Loopback1
  prefix-attributes anycast
  address-family ipv4 unicast
   prefix-sid absolute 16012 // Anycast SID (prefer any)
  !
 !
 interface Loopback2
  prefix-attributes anycast
  address-family ipv4 unicast
   metric 100000
   prefix-sid absolute 17012 // Weighted Anycast SID (prefer node 1)
  !
 !
 interface Loopback3
  prefix-attributes anycast
  address-family ipv4 unicast
   metric 1
   prefix-sid absolute 18012 // Weighted Anycast SID (prefer node 2)
  !
 !
!
end
```

# Configuring an Adjacency SID

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

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

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

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

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

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

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|F|B|V|L|S|P|   |
+-+-+-+-+-+-+-+-+
```

*Table 12: Adjacency Segment Identifier (Adj-SID) Flags Sub-TLV Fields*

| Field | Description |
|---|---|
| S (Set) | This flag is set if the same Adj-SID value has been provisioned on multiple interfaces. |
| P (Persistent) | This flag is set if the Adj-SID is persistent (manually allocated). |

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.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>• You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command. |
| **Step 3** | **interface** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis)# interface GigabitEthernet0/0/0/7` | Specifies the interface and enters interface configuration mode. |
| **Step 4** | **point-to-point**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if)# point-to-point` | Specifies the interface is a point-to-point interface. |
| **Step 5** | **address-family** { **ipv4** \| **ipv6** } [ **unicast** ]<br>**Example:**<br>The following is an example for ipv4 address family:<br>`RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast` | Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |
| **Step 6** | **adjacency-sid** {**index** *adj-SID-index* \| **absolute** *adj-SID-value* } [**protected** ]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if-af)# adjacency-sid index 10`<br>`RP/0/RP0/CPU0:router(config-isis-if-af)# adjacency-sid absolute 15010` | Configures the Adj-SID index or absolute value for the interface.<br>Specify **index** *adj-SID-index* for each link to create an Ajd-SID based on the lower boundary of the SRLB + the index.<br>Specify **absolute** *adj-SID-value* for each link to create a specific Ajd-SID within the SRLB.<br>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, |

| | Command or Action | Purpose |
|---|---|---|
| | | a backup path is installed. By default, manual Adj-SIDs are not protected. |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Verify the Adj-SID configuration:

```
RP/0/RP0/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/RP0/CPU0:router# show mpls forwarding labels 15010
Mon Jun 12 02:50:12.172 PDT
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
15010  Pop         SRLB (idx 10)      Gi0/0/0/3    10.0.3.3        0
       Pop         SRLB (idx 10)      Gi0/0/0/7    10.1.0.5        0
       16004       SRLB (idx 10)      Gi0/0/0/7    10.1.0.5        0           (!)
       16004       SRLB (idx 10)      Gi0/0/0/3    10.0.3.3        0           (!)
```

# Manually Configure 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 3 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 3 bundle link goes up and down.
- Manually configured Layer 2 Adj-SIDs are persistent if the Layer 3 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.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **segment-routing**<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config)# segment-routing` | Enters segment routing configuration mode. |
| **Step 3** | **adjacency-sid**<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config-sr)# adjacency-sid` | Enters adjacency SID configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **interface** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config-sr-adj)# interface GigabitEthernet0/0/0/3` | Specifies the interface and enters interface configuration mode. |
| **Step 5** | **address-family** { **ipv4** \| **ipv6** } [ **unicast** ]<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config-sr-adj-intf)# address-family ipv4 unicast` | Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |
| **Step 6** | **l2-adjacency sid** {**index** *adj-SID-index* \| **absolute** *adj-SID-value* } [**next-hop** { *ipv4_address* \| *ipv6_address* } ]<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config-sr-adj-intf-af)# l2-adjacency sid absolute 15015 next-hop 10.1.1.4` | Configures the Adj-SID index or absolute value for the interface.<br>Specify **index** *adj-SID-index* for each link to create an Ajd-SID based on the lower boundary of the SRLB + the index.<br>Specify **absolute** *adj-SID-value* for each link to create a specific Ajd-SID within the SRLB.<br>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.<br>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. |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 8** | **end** | |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config)# router isis isp` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode. |
| **Step 10** | **address-family** { **ipv4** \| **ipv6** } [ **unicast** ]<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config-isis)# address-family ipv4 unicast` | Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |
| **Step 11** | **segment-routing bundle-member-adj-sid**<br>**Example:**<br>`RP/0/RP0/CPU0:Router(config-isis-af)# segment-routing bundle-member-adj-sid` | Programs the dynamic Layer 2 Adj-SIDs, and advertises both manual and dynamic Layer 2 Adj-SIDs.<br>**Note** This command is not required to program manual L2 Adj-SID, but is required to program the dynamic Layer 2 Adj-SIDs and to advertise both manual and dynamic Layer 2 Adj-SIDs. |

Verify the configuration:

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


Router# show running-config segment-routing
Tue Jun 20 07:14:25.815 PDT
segment-routing
 adjacency-sid
  interface GigabitEthernet0/0/0/3
   address-family ipv4 unicast
    l2-adjacency-sid absolute 15015 next-hop 10.1.1.4
   !
  !
 !
!
```

# 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

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>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** ]<br>**Example:**<br>The following is an example for ipv4 address family:<br>`RP/0/RP0/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**<br>**Example:**<br>`RP/0/RP0/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. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session. |

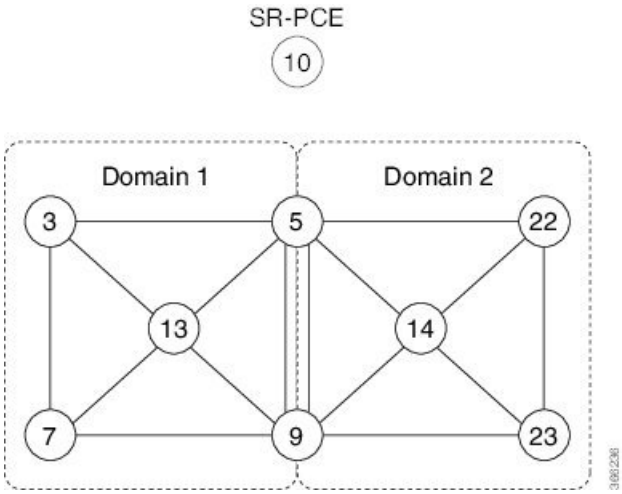
| | Command or Action | Purpose |
|---|---|---|
| | | • **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# 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).

***Figure 9: Multi-Domain Topology***




## Configure IS-IS Multi-Domain Prefix SID

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

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

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

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

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

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

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

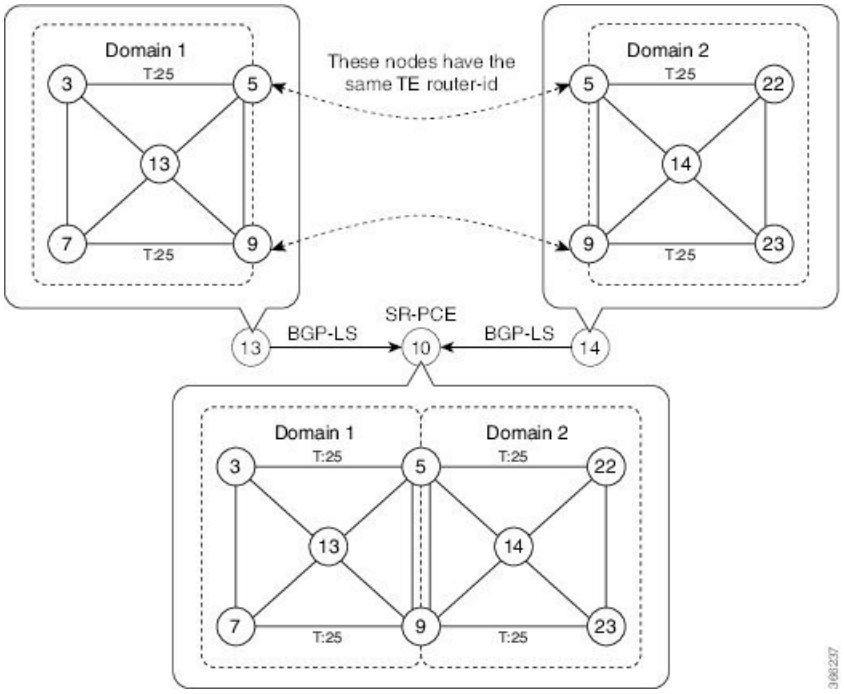
# Configure Common Router ID

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

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

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

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

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

# Distribute IS-IS Link-State Data




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

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

```
Example: Node 14
router isis Domain2
 distribute link-state instance-id instance-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.

# Conditional Prefix Advertisement

In some situations, it's beneficial to make the IS-IS prefix advertisement conditional. For example, an Area Border Router (ABR) or Autonomous System Boundary Router (ASBR) that has lost its connection to one of the areas or autonomous systems (AS) might keep advertising a prefix. If an ABR or ASBR advertises the Segment Routing (SR) SID with this prefix, the label stack of the traffic routed toward the disconnected area or AS might use this SID, which would result in dropped traffic at the ABR or ASBR.

ABRs or ASBRs are often deployed in pairs for redundancy and advertise a shared Anycast prefix SID. Conditional Prefix Advertisement allows an ABR or an ASBR to advertise its Anycast SID only when connected to a specific area or domain. If an ABR or ASBR becomes disconnected from the particular area or AS, it stops advertising the address for a specified interface (for example, Loopback).

Configure the conditional prefix advertisement under a specific interface. The prefix advertisement on this interface is associated with the route-policy that tracks the presence of a set of prefixes (prefix-set) in the Routing Information Base (RIB).

For faster convergence, the route-policy used for conditional prefix advertisement uses the new event-based **rib-has-route async** condition to notify IS-IS of the following situations:

- When the last prefix from the prefix-set is removed from the RIB.
- When the first prefix from the prefix-set is added to the RIB.

### Configuration

To use the conditional prefix advertisement in IS-IS, create a prefix-set to be tracked. Then create a route policy that uses the prefix-set.

```
Router(config)# prefix-set prefix-set-name
Router(config-pfx)# prefix-address-1/length[, prefix-address-2/length,,,
prefix-address-16/length]
Router(config-pfx)# end-set

Router(config)# route-policy rpl-name
Router(config-rpl)# if rib-has-route async prefix-set-name then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
```

To advertise the loopback address in IS-IS conditionally, use the **advertise prefix route-policy** command under IS-IS interface address-family configuration sub-mode.

```
Router(config)# router isis 1
Router(config-isis)# interface Loopback0
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# advertise prefix route-policy rpl-name
Router(config-isis-if-af)# commit
```

### Example

```
Router(config)# prefix-set domain_2
Router(config-pfx)# 2.3.3.3/32, 2.4.4.4/32
Router(config-pfx)# end-set
Router(config)# route-policy track_domain_2
Router(config-rpl)# if rib-has-route async domain_2 then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
Router(config)# router isis 1
Router(config-isis)# interface Loopback0
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# advertise prefix route-policy track_domain-2
Router(config-isis-if-af)# commit
```

### Running Configuration

```
prefix-set domain_2
  2.3.3.3/32,
  2.4.4.4/32
end-set
!
route-policy track_domain_2
  if rib-has-route async domain_2 then
    pass
  endif
end-policy
!
router isis 1
 interface Loopback0
  address-family ipv4 unicast
   advertise prefix route-policy track_domain_2
  !
 !
!
```

# Segment Routing ECMP-FEC Optimization

*Table 13: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR ECMP-FEC Optimization L2 and L3 Recursive Services | Release 7.4.1 | This feature adds support for L2VPN service Label Edge Router (LER) and BGP PIC for Layer 3 BGP services when SR ECMP-FEC Optimization is enabled. |

ECMP-FECs are used for any ECMP programming on the system, such as MPLS LSP ECMP, VPN multipath, and EVPN multi-homing.

The SR ECMP-FEC optimization solution minimizes ECMP-FEC resource consumption and duplication, during underlay programming for an SR-MPLS network. This feature supports sharing the same ECMP-FEC, regular FEC, and Egress Encapsulation DB (EEDB) entries among Segment Routing prefixes with the same set of next hops.

ECMP-FEC optimization is triggered when all the out_labels associated with the ECMP paths for a given prefix have the same value. If this rule is not met, then the prefix is programmed with a dedicated ECMP-FEC.

Segment Routing Label Edge Router (LER) ECMP-FEC Optimization enables ECMP-FEC optimization originally developed for Label Switched Router (LSR) nodes (MPLS P) to be enabled on LER (Layer 3 MPLS PE) routers.

### Usage Guidelines and Limitations

- For the labeled prefixes with ECMP across a combination of labeled and unlabeled (PHP) paths, the SR ECMP-FEC Optimization cannot be triggered since the paths associated with the prefix do not have the same outgoing label and/or label action.

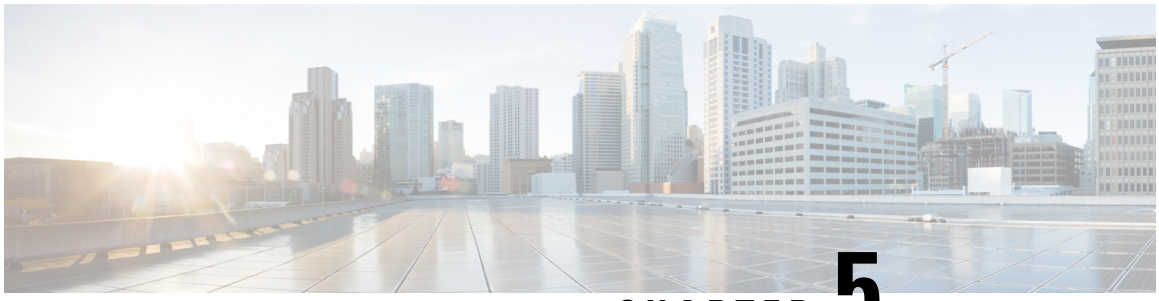
- For prefixes with LFA backup paths, the SR ECMP-FEC Optimization is possible since these backup paths do not require an extra label to be pushed; all paths associated with the prefix (primary and backup) have the same outgoing label value.
- For prefixes with TI-LFA backup paths requiring extra labels to be pushed on to the backup, the SR ECMP-FEC Optimization is not possible since all the paths associated with the prefix do not have the same outgoing label value.
- For the duration of time that prefixes are programmed to avoid microloops (when SR MicroLoop Avoidance is triggered), SR ECMP-FEC Optimization is not possible since all the paths associated with the prefix do not have the same outgoing label value. After removal of the microloop-avoidance programming, the SR ECMP-FEC Optimization might be possible again.
- For scenarios with prefixes where the SR ECMP-FEC Optimization is not possible, dedicated ECMP-FEC is allocated per prefix. This could potentially lead to ECMP FEC out-of-resource (OOR) considering the baseline usage of ECMP FEC resources at steady state. During ECMP-FEC OOR, prefixes with multiple paths are programmed with a single path in order to avoid traffic disruption.
- SR ECMP-FEC optimization is applicable in the following instances:
  - Label Switched Router (LSR) nodes (MPLS P)
  - L3VPN Label Edge Router (LER) nodes
  - L2VPN LER nodes
  - ASBR node with BGP-LU swap
- BGP PIC is supported
- SR ECMP-FEC optimization should not be enabled in the following instances:
  - L2VPN LER nodes
  - L2VPN/L3VPN LER nodes with VPN over BGP-LU over SR
- BGP PIC is not supported.
- For the labeled prefixes, transitioning from TI-LFA to SR ECMP-FEC optimization can cause ECMP-FEC OOR due to different output labels (ECMP label vs backup path's label) at make-before-break. This results in a few second traffic loss depending on route scale

### Enable SR ECMP-FEC Optimization

To enable SR ECMP-FEC optimization, use the **hw-module fib mpls label lsr-optimized** command in global configuration mode. After enabling this feature, reload the line card. For more information about the command, see the *MPLS Label Distribution Protocol Commands* chapter in the *MPLS Command Reference Guide*.

```
Router(config)# hw-module fib mpls label lsr-optimized
Router(config)# commit

LC/0/0/CPU0:Oct 11 20:19:12.540 UTC: fia_driver[185]:
%FABRIC-FIA_DRVR-4-MPLS_HW_PROFILE_MISMATCH :
    Mismatch found, reload LC to activate the new mpls profile


Router# reload location 0/0/CPU0
```

```
Proceed with reload? [confirm]
Reloading node 0/0/CPU0
```



CHAPTER 5

# Configure Segment Routing for OSPF Protocol

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

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

**Note** For additional information on implementing OSPF on your , see the *Implementing OSPF* module in the .



# 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.

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

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router ospf** *process-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router ospf 1` | Enables OSPF routing for the specified routing process and places the router in router configuration mode. |
| **Step 3** | **segment-routing mpls**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# segment-routing mpls` | Enables segment routing using the MPLS data plane on the routing process and all areas and interfaces in the routing process.<br>Enables segment routing fowarding on all interfaces in the routing process and installs the SIDs received by OSPF in the forwarding table. |
| **Step 4** | **segment-routing sr-prefer**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# segment-routing sr-prefer` | Sets the preference of segment routing (SR) labels over label distribution protocol (LDP) labels. |
| **Step 5** | **area** *area*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# area 0` | Enters area configuration mode. |
| **Step 6** | **segment-routing mpls**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar)# segment-routing mpls` | (Optional) Enables segment routing using the MPLS data plane on the area and all interfaces in the area. Enables segment routing fowarding on all interfaces in the area and installs the SIDs received by OSPF in the forwarding table. |
| **Step 7** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar)# exit`<br>`RP/0/RP0/CPU0:router(config-ospf)# exit` | |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

Configure the prefix SID.

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

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

A prefix SID can be a node SID or an Anycast SID. A node SID is a type of prefix SID that identifies a specific node. An Anycast SID is a type of prefix SID that identifies a set of nodes, and is configured with n-flag clear. The set of nodes (Anycast group) is configured to advertise a shared prefix address and prefix SID. Anycast routing enables the steering of traffic toward multiple advertising nodes. Packets addressed to an Anycast address are forwarded to the topologically nearest nodes.

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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:** | Enters mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# configure` | |
| **Step 2** | **router ospf** *process-name*<br>**Example:**<br>`RP/0/RP0/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** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# area 0` | Enters area configuration mode. |
| **Step 4** | **interface Loopback** *interface-instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar)# interface loopback 0` | Specifies the loopback interface and instance. |
| **Step 5** | **prefix-sid**{**index** *SID-index* \| **absolute** *SID-value* } [**n-flag-clear**] [**explicit-null**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar)# prefix-sid index 1001`<br>`RP/0/RP0/CPU0:router(config-ospf-ar)# prefix-sid absolute 17001` | Configures the prefix-SID index or absolute value for the interface.<br>Specify **index** *SID-index* for each node to create a prefix SID based on the lower boundary of the SRGB + the index.<br>Specify **absolute** *SID-value* for each node to create a specific prefix SID within the SRGB.<br>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).<br>To disable penultimate-hop-popping (PHP) and add an explicit-Null label, enter the `explicit-null` keyword. OSPF sets the `E` flag in the prefix-SID sub TLV. |
| **Step 6** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Verify the prefix-SID configuration:

```
RP/0/RP0/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
```

# Conditional Prefix Advertisement

In some situations, it's beneficial to make the OSPF prefix advertisement conditional. For example, an Area Border Router (ABR) or Autonomous System Boundary Router (ASBR) that has lost its connection to one of the areas or autonomous systems (AS) might keep advertising a prefix. If an ABR or ASBR advertises the Segment Routing (SR) SID with this prefix, the label stack of the traffic routed toward the disconnected area or AS might use this SID, which would result in dropped traffic at the ABR or ASBR.

ABRs or ASBRs are often deployed in pairs for redundancy and advertise a shared Anycast prefix SID. Conditional Prefix Advertisement allows an ABR or an ASBR to advertise its Anycast SID only when connected to a specific area or domain. If an ABR or ASBR becomes disconnected from the particular area or AS, it stops advertising the address for a specified interface (for example, Loopback).

Configure the conditional prefix advertisement under a specific interface. The prefix advertisement on this interface is associated with the route-policy that tracks the presence of a set of prefixes (prefix-set) in the Routing Information Base (RIB).

For faster convergence, the route-policy used for conditional prefix advertisement uses the new event-based **rib-has-route async** condition to notify OSPF of the following situations:

- When the last prefix from the prefix-set is removed from the RIB.
- When the first prefix from the prefix-set is added to the RIB.

### Configuration

To use the conditional prefix advertisement in OSPF, create a prefix-set to be tracked. Then create a route policy that uses the prefix-set.

```
Router(config)# prefix-set prefix-set-name
Router(config-pfx)# prefix-address-1/length[, prefix-address-2/length,,,
prefix-address-16/length]
Router(config-pfx)# end-set

Router(config)# route-policy rpl-name
Router(config-rpl)# if rib-has-route async prefix-set-name then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
```

To advertise the loopback address in OSPF conditionally, use the **advertise prefix route-policy** command under OSPF interface address-family configuration sub-mode.

```
Router(config)# router ospf 1
Router(config-ospf)# area 0
Router(config-ospf-ar)# interface Loopback0
Router(config-ospf-ar-if)# advertise prefix route-policy rpl-name
Router(config-ospf-ar-if)# commit
```

### Example

```
Router(config)# prefix-set domain_2
Router(config-pfx)# 2.3.3.3/32, 2.4.4.4/32
Router(config-pfx)# end-set
Router(config)# route-policy track_domain_2
Router(config-rpl)# if rib-has-route async domain_2 then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
Router(config)# router ospf 1
Router(config-ospf)# area 0
Router(config-ospf-ar)# interface Loopback0
Router(config-ospf-ar-if)# advertise prefix route-policy track_domain-2
Router(config-ospf-ar-if)# commit
```

### Running Configuration

```
prefix-set domain_2
  2.3.3.3/32,
  2.4.4.4/32
end-set
!
route-policy track_domain_2
  if rib-has-route async domain_2 then
    pass
  endif
end-policy
!
router ospf 1
 area 0
  interface Loopback0
   advertise prefix route-policy track_domain_2
  !
 !
!
```

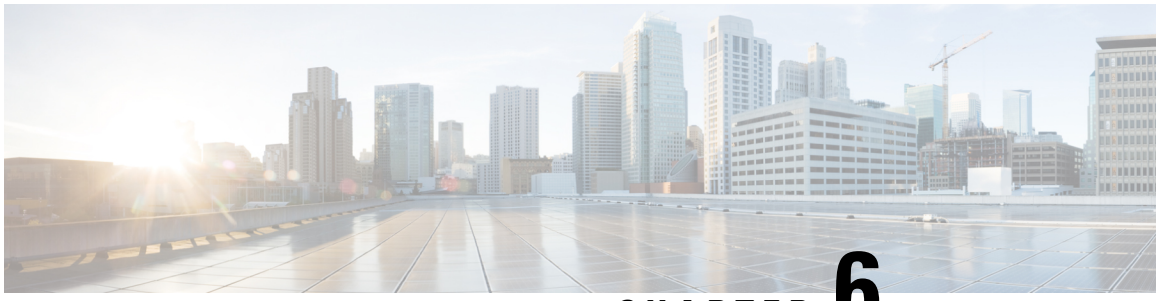
# Segment Routing ECMP-FEC Optimization

ECMP-FECs are used for any ECMP programming on the system, such as MPLS LSP ECMP, VPN multipath, and EVPN multi-homing.

The SR ECMP-FEC optimization solution minimizes ECMP-FEC resource consumption during underlay programming for an SR-MPLS network. This feature supports sharing the same ECMP-FEC, regular FEC, and Egress Encapsulation DB (EEDB) entries for all IPv4 and IPv6 Segment Routing prefixes with the same set of next hops. ECMP-FEC optimization is triggered when all the out_labels associated with the ECMP paths for a given prefix have the same value. If this rule is not met, then the prefix is programmed with a dedicated ECMP-FEC. Other prefixes that meet the rule are candidates for optimization.

Segment Routing Label Edge Router (LER) ECMP-FEC Optimization enables ECMP-FEC optimization originally developed for Label Switched Router (LSR) nodes (MPLS P) to be enabled on LER (Layer 3 MPLS PE) routers.

For usage guidelines, limitations, and configuration options, see Segment Routing ECMP-FEC Optimization, on page 127.



CHAPTER 6

# Configure Segment Routing for BGP

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

This module provides the configuration information used to enable Segment Routing for BGP.

**Note** For additional information on implementing BGP on your router , see the *Implementing BGP* module in the *Routing Configuration Guide for Cisco NCS 540 Series Routers*.



## Segment Routing for BGP

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

In an MPLS-based DC fabric, the eBGP sessions between the nodes exchange BGP labeled unicast (BGP-LU) network layer reachability information (NLRI). An MPLS-based DC fabric allows any leaf (top-of-rack or border router) in the fabric to communicate with any other leaf using a single label, which results in higher packet forwarding performance and lower encapsulation overhead than traditional BGP-based DC fabric. However, since each label value might be different for each hop, an MPLS-based DC fabric is more difficult to troubleshoot and more complex to configure.

BGP has been extended to carry segment routing prefix-SID index. BGP-LU helps each node learn BGP prefix SIDs of other leaf nodes and can use ECMP between source and destination. Segment routing for BGP

simplifies the configuration, operation, and troubleshooting of the fabric. With segment routing for BGP, you can enable traffic steering capabilities in the data center using a BGP prefix SID.

**Note** BGP flowspec support with SRv6 - Limitations

List of BGP address families interacts with SRv6. There are some supported and unsupported BGP address family for the interaction with SRv6.

Supported address family:

- address-families ipv6.

Unsupported address families:

- address-families ipv4
- vpnv4
- vpnv6

# 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** You must enable SR and explicitly configure the SRGB before configuring SR BGP. The SRGB must be explicitly configured, even if you are using the default range (16000 – 23999). BGP uses the SRGB and the index in the BGP prefix-SID attribute of a learned BGP-LU advertisement to allocate a local label for a given destination.

If SR and the SRGB are enabled after configuring BGP, then BGP is not aware of the SRGB, and therefore it allocates BGP-LU local labels from the dynamic label range instead of from the SRGB. In this case, restart the BGP process in order to allocate BGP-LU local labels from 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 *Routing Configuration Guide for Cisco NCS 540 Series Routers*.

### 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/RP0/CPU0:router(config)# segment-routing global-block 16000 23999

RP/0/RP0/CPU0:router(config)# route-policy SID($SID)
RP/0/RP0/CPU0:router(config-rpl)# set label-index $SID
RP/0/RP0/CPU0:router(config-rpl)# end policy

RP/0/RP0/CPU0:router(config)# router bgp 1
RP/0/RP0/CPU0:router(config-bgp)# bgp router-id 10.1.1.1
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# network 10.1.1.3/32 route-policy SID(3)
RP/0/RP0/CPU0:router(config-bgp-af)# allocate-label all
RP/0/RP0/CPU0:router(config-bgp-af)# commit
RP/0/RP0/CPU0:router(config-bgp-af)# end


RP/0/RP0/CPU0:router# show bgp 10.1.1.3/32
BGP routing table entry for 10.1.1.3/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                74          74
    Local Label: 16003
Last Modified: Sep 29 19:52:18.155 for 00:07:22
Paths: (1 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
  Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.2
  3
    99.3.21.3 from 99.3.21.3 (10.1.1.3)
      Received Label 3
      Origin IGP, metric 0, localpref 100, valid, external, best, group-best
      Received Path ID 0, Local Path ID 1, version 74
      Origin-AS validity: not-found
      Label Index: 3
```

# 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.

Below are the BGP-EPE peering SID types:

- PeerNode SID—To an eBGP peer. Pops the label and forwards the traffic on any interface to the peer.
- PeerAdjacency SID—To an eBGP peer via interface. Pops the label and forwards the traffic on the related interface.
- PeerSet SID—To a set of eBGP peers. Pops the label and forwards the traffic on any interface to the set of peers. All the peers in a set might not be in the same AS.

  Multiple PeerSet SIDs can be associated with any combination of PeerNode SIDs or PeerAdjacency SIDs.

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.

# Configure Segment Routing Egress Peer Engineering

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **router bgp** *as-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router bgp 1` | Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process. |
| **Step 2** | **neighbor** *ip-address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp)# neighbor 192.168.1.3` | Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer. |
| **Step 3** | **remote-as** *as-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 3` | Creates a neighbor and assigns a remote autonomous system number to it. |
| **Step 4** | **egress-engineering**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp-nbr)#` | Configures the egress node with EPE for the eBGP peer. |

| | Command or Action | Purpose |
|---|---|---|
| | **egress-engineering** | |
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp-nbr)# exit`<br>`RP/0/RP0/CPU0:router(config-bgp)# exit`<br>`RP/0/RP0/CPU0:router(config)#` | |
| **Step 6** | **mpls static**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# mpls static` | Configure MPLS static on the egress interface connecting to the eBGP peer. |
| **Step 7** | **interface** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpls-static)# interface GigabitEthernet0/0/1/2` | Specifies the egress interface connecting to the eBGP peer. |
| **Step 8** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### Example

**Running Config:**

```
router bgp 1
 neighbor 192.168.1.3
  remote-as 3
  egress-engineering
  !
 !
```

```
mpls static
 interface GigabitEthernet0/0/1/2
 !
!
```

# Configuring Manual BGP-EPE Peering SIDs

Configuring manual BGP-EPE Peer SIDs allows for persistent EPE label values. Manual BGP-EPE SIDs are advertised through BGP-LS and are allocated from the Segment Routing Local Block (SRLB). See Configure Segment Routing Global Block and Segment Routing Local Block, on page 97 for information about the SRLB.

Each PeerNode SID, PeerAdjacency SID, and PeerSet SID is configured with an index value. This index serves as an offset from the configured SRLB start value and the resulting MPLS label (SRLB start label + index) is assigned to these SIDs. This label is used by CEF to perform load balancing across the individual BGP PeerSet SIDs, BGP PeerNode SID, or ultimately across each first-hop adjacency associated with that BGP PeerNode SID or BGP PeerSet SID.

### Configuring Manual PeerNode SID

Each eBGP peer will be associated with a PeerNode SID index that is configuration driven.

```
RP/0/0/CPU0:PE1(config)# router bgp 10
RP/0/0/CPU0:PE1(config-bgp)# neighbor 10.10.10.2
RP/0/0/CPU0:PE1(config-bgp-nbr)# remote-as 20
RP/0/0/CPU0:PE1(config-bgp-nbr)# egress-engineering
RP/0/0/CPU0:PE1(config-bgp-nbr)# peer-node-sid index 600
```

### Configuring Manual PeerAdjacency SID

Any first-hop for which an adjacency SID is configured needs to be in the resolution chain of at least one eBGP peer that is configured for egress-peer engineering. Otherwise such a kind of “orphan” first-hop with regards to BGP has no effect on this feature. This is because BGP only understands next-hops learnt by the BGP protocol itself and in addition only the resolving IGP next-hops for those BGP next-hops.

```
RP/0/0/CPU0:PE1(config)# router bgp 10
RP/0/0/CPU0:PE1(config-bgp)# adjacencies
RP/0/0/CPU0:PE1(config-bgp-adj)# 10.1.1.2
RP/0/0/CPU0:PE1(config-bgp-adj)# adjacency-sid index 500
```

### Configuring Manual PeerSet SID

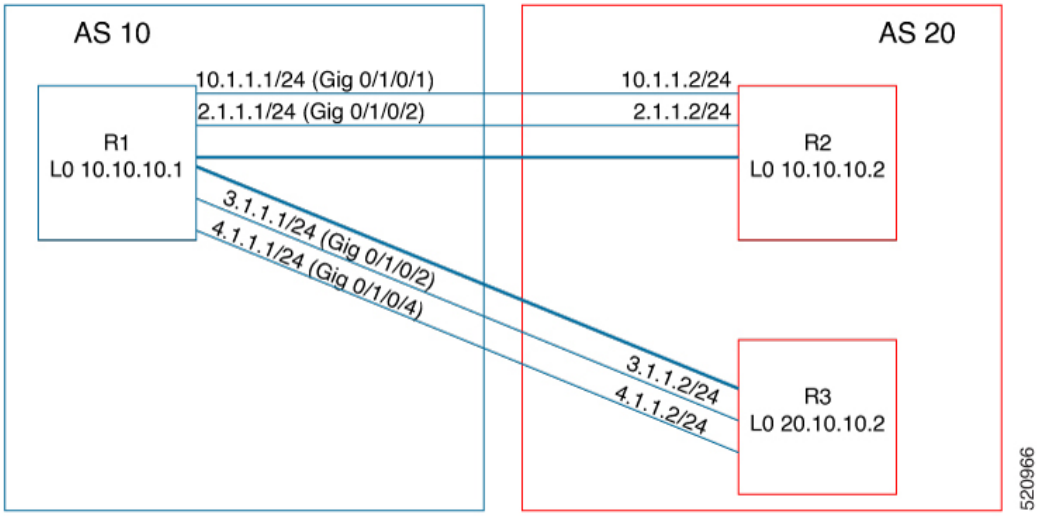
The PeerSet SID is configured under global Address Family. This configuration results in the creation of a Peer-Set SID EPE object.

```
RP/0/0/CPU0:PE1(config)# router bgp 10
RP/0/0/CPU0:PE1(config-bgp)# address-family ipv4 unicast
RP/0/0/CPU0:PE1(config-bgp-afi)# peer-set-id 1
RP/0/0/CPU0:PE1(config-bgp-peer-set)# peer-set-sid 300
```

### Example

#### Topology

The example in this section uses the following topology.




In this example, BGP-EPE peer SIDs are allocated from the default SRLB label range (15000 – 15999). The BGP-EPE peer SIDs are configured as follows:

- PeerNode SIDs to 10.10.10.2 with index 600 (label 15600), and for 20.10.10.2 with index 700 (label 15700)
- PeerAdj SID to link 10.1.1.2 with index 500 (label 15500)
- PeerSet SID 1 to load balance over BGP neighbors 10.10.10.1 and 20.10.10.2 with SID index 300 (label 15300)
- PeerSet SID 2 to load balance over BGP neighbor 20.10.10.2 and link 10.1.1.2 with SID index 400 (label 15400)

### Configuration on R1

```
router bgp 10
 address-family ipv4 unicast
  peer-set-id 1
   peer-set-sid index 300
  !
  peer-set-id 2
   peer-set-sid index 400
  !
 !
 adjacencies
  10.1.1.2
   adjacency-sid index 500
   peer-set 2
  !
 !
 neighbor 10.10.10.2
  remote-as 20
  egress-engineering
  peer-node-sid index 600
  peer-set 1
 !
 neighbor 20.10.10.2
  egress-engineering
  peer-node-sid index 700
```

```
 peer-set 1
 peer-set 2
!
```

To further show the load balancing of this example:

- 15600 is load balanced over {10.1.1.1 and 2.1.1.1}
- 15700 is load balanced over {3.1.1.1 and 4.1.1.1}
- 15500 is load balanced over {10.1.1.1}
- 15300 is load balanced over {10.1.1.1, 2.1.1.1, 3.1.1.1 and 4.1.1.1}
- 15400 is load balanced over {10.1.1.1, 3.1.1.1 and 4.1.1.1}

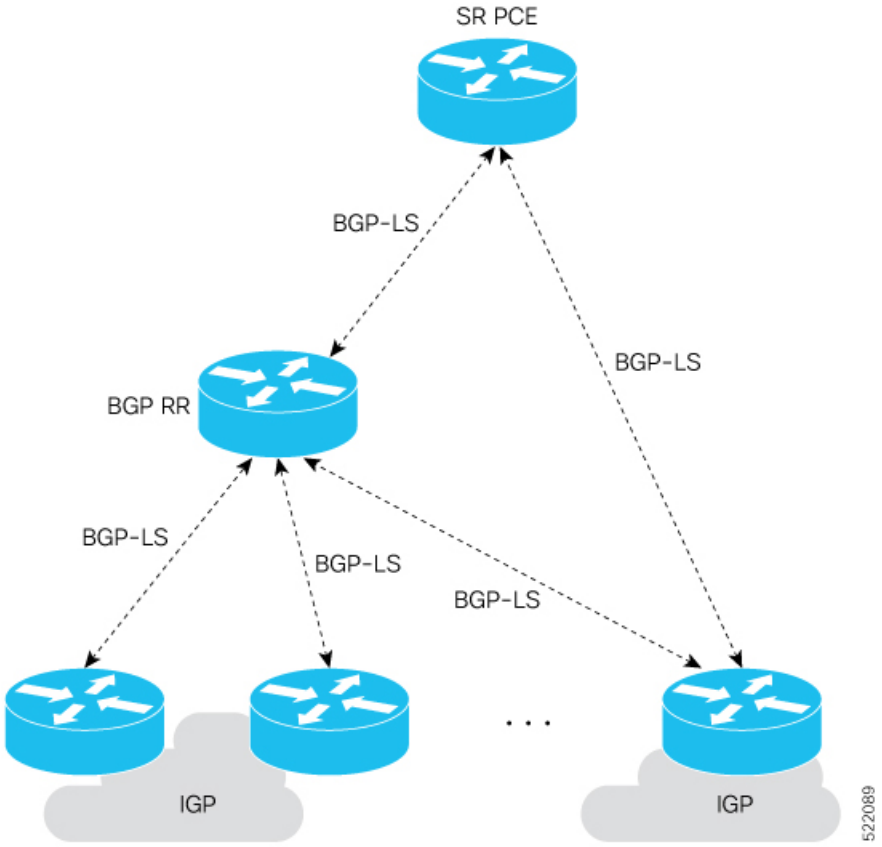
# Configure BGP Link-State

BGP Link-State (LS) is an Address Family Identifier (AFI) and Sub-address Family Identifier (SAFI) originally defined to carry interior gateway protocol (IGP) link-state information through BGP. The BGP Network Layer Reachability Information (NLRI) encoding format for BGP-LS and a new BGP Path Attribute called the BGP-LS attribute are defined in RFC7752. The identifying key of each Link-State object, namely a node, link, or prefix, is encoded in the NLRI and the properties of the object are encoded in the BGP-LS attribute.

The BGP-LS Extensions for Segment Routing are documented in RFC9085.

BGP-LS applications like an SR Path Computation Engine (SR-PCE) can learn the SR capabilities of the nodes in the topology and the mapping of SR segments to those nodes. This can enable the SR-PCE to perform path computations based on SR-TE and to steer traffic on paths different from the underlying IGP-based distributed best-path computation.

The following figure shows a typical deployment scenario. In each IGP area, one or more nodes (BGP speakers) are configured with BGP-LS. These BGP speakers form an iBGP mesh by connecting to one or more route-reflectors. This way, all BGP speakers (specifically the route-reflectors) obtain Link-State information from all IGP areas (and from other ASes from eBGP peers).




### Usage Guidelines and Limitations

- BGP-LS supports IS-IS and OSPFv2.
- The identifier field of BGP-LS (referred to as the Instance-ID) identifies the IGP routing domain where the NLRI belongs. The NLRIs representing link-state objects (nodes, links, or prefixes) from the same IGP routing instance must use the same Instance-ID value.
- When there is only a single protocol instance in the network where BGP-LS is operational, we recommend configuring the Instance-ID value to **0**.
- Assign consistent BGP-LS Instance-ID values on all BGP-LS Producers within a given IGP domain.
- NLRIs with different Instance-ID values are considered to be from different IGP routing instances.
- Unique Instance-ID values must be assigned to routing protocol instances operating in different IGP domains. This allows the BGP-LS Consumer (for example, SR-PCE) to build an accurate segregated multi-domain topology based on the Instance-ID values, even when the topology is advertised via BGP-LS by multiple BGP-LS Producers in the network.
- If the BGP-LS Instance-ID configuration guidelines are not followed, a BGP-LS Consumer may see duplicate link-state objects for the same node, link, or prefix when there are multiple BGP-LS Producers deployed. This may also result in the BGP-LS Consumers getting an inaccurate network-wide topology.

- The following table defines the supported extensions to the BGP-LS address family for carrying IGP topology information (including SR information) via BGP. For more information on the BGP-LS TLVs, refer to Border Gateway Protocol - Link State (BGP-LS) Parameters.

*Table 14: IOS XR Supported BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs*

| TLV Code Point | Description | Produced by IS-IS | Produced by OSPFv2 | Produced by BGP |
|---|---|---|---|---|
| 256 | Local Node Descriptors | X | X | — |
| 257 | Remote Node Descriptors | X | X | — |
| 258 | Link Local/Remote Identifiers | X | X | — |
| 259 | IPv4 interface address | X | X | — |
| 260 | IPv4 neighbor address | X | | |
| 261 | IPv6 interface address | X | — | — |
| 262 | IPv6 neighbor address | X | — | — |
| 263 | Multi-Topology ID | X | — | — |
| 264 | OSPF Route Type | — | X | — |
| 265 | IP Reachability Information | X | X | — |
| 266 | Node MSD TLV | X | X | — |
| 267 | Link MSD TLV | X | X | — |
| 512 | Autonomous System | — | — | X |
| 513 | BGP-LS Identifier | — | — | X |
| 514 | OSPF Area-ID | — | X | — |
| 515 | IGP Router-ID | X | X | — |
| 516 | BGP Router-ID TLV | — | — | X |
| 517 | BGP Confederation Member TLV | — | — | X |
| 1024 | Node Flag Bits | X | X | — |
| 1026 | Node Name | X | X | — |
| 1027 | IS-IS Area Identifier | X | — | — |
| 1028 | IPv4 Router-ID of Local Node | X | X | — |
| 1029 | IPv6 Router-ID of Local Node | X | — | — |
| 1030 | IPv4 Router-ID of Remote Node | X | X | — |
| 1031 | IPv6 Router-ID of Remote Node | X | — | — |
| 1034 | SR Capabilities TLV | X | X | — |
| 1035 | SR Algorithm TLV | X | X | — |
| 1036 | SR Local Block TLV | X | X | — |

| TLV Code Point | Description | Produced by IS-IS | Produced by OSPFv2 | Produced by BGP |
|---|---|---|---|---|
| 1039 | Flex Algo Definition (FAD) TLV | X | X | — |
| 1044 | Flex Algorithm Prefix Metric (FAPM) TLV | X | X | — |
| 1088 | Administrative group (color) | X | X | — |
| 1089 | Maximum link bandwidth | X | X | — |
| 1090 | Max. reservable link bandwidth | X | X | — |
| 1091 | Unreserved bandwidth | X | X | — |
| 1092 | TE Default Metric | X | X | — |
| 1093 | Link Protection Type | X | X | — |
| 1094 | MPLS Protocol Mask | X | X | — |
| 1095 | IGP Metric | X | X | — |
| 1096 | Shared Risk Link Group | X | X | — |
| 1099 | Adjacency SID TLV | X | X | — |
| 1100 | LAN Adjacency SID TLV | X | X | — |
| 1101 | PeerNode SID TLV | — | — | X |
| 1102 | PeerAdj SID TLV | — | — | X |
| 1103 | PeerSet SID TLV | — | — | X |
| 1114 | Unidirectional Link Delay TLV | X | X | — |
| 1115 | Min/Max Unidirectional Link Delay TLV | X | X | — |
| 1116 | Unidirectional Delay Variation TLV | X | X | — |
| 1117 | Unidirectional Link Loss | X | X | — |
| 1118 | Unidirectional Residual Bandwidth | X | X | — |
| 1119 | Unidirectional Available Bandwidth | X | X | — |
| 1120 | Unidirectional Utilized Bandwidth | X | X | — |
| 1122 | Application-Specific Link Attribute TLV | X | X | — |
| 1152 | IGP Flags | X | X | — |
| 1153 | IGP Route Tag | X | X | — |
| 1154 | IGP Extended Route Tag | X | — | — |
| 1155 | Prefix Metric | X | X | — |
| 1156 | OSPF Forwarding Address | — | X | — |
| 1158 | Prefix-SID | X | X | — |
| 1159 | Range | X | X | — |

| TLV Code Point | Description | Produced by IS-IS | Produced by OSPFv2 | Produced by BGP |
|---|---|---|---|---|
| 1161 | SID/Label TLV | X | X | — |
| 1170 | Prefix Attribute Flags | X | X | — |
| 1171 | Source Router Identifier | X | — | — |
| 1172 | L2 Bundle Member Attributes TLV | X | — | — |
| 1173 | Extended Administrative Group | X | X | — |

### Exchange Link State Information with BGP Neighbor

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

```
Router# configure
Router(config)# router bgp 1
Router(config-bgp)# neighbor 10.0.0.2
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# address-family link-state link-state
Router(config-bgp-nbr-af)# exit
```

### IGP Link-State Database Distribution

A given BGP node may have connections to multiple, independent routing domains. IGP link-state database distribution into BGP-LS is supported for both OSPF and IS-IS protocols in order to distribute this information on to controllers or applications that desire to build paths spanning or including these multiple domains.

To distribute IS-IS link-state data using BGP-LS, use the **distribute link-state** command in router configuration mode.

```
Router# configure
Router(config)# router isis isp
Router(config-isis)# distribute link-state instance-id 32
```

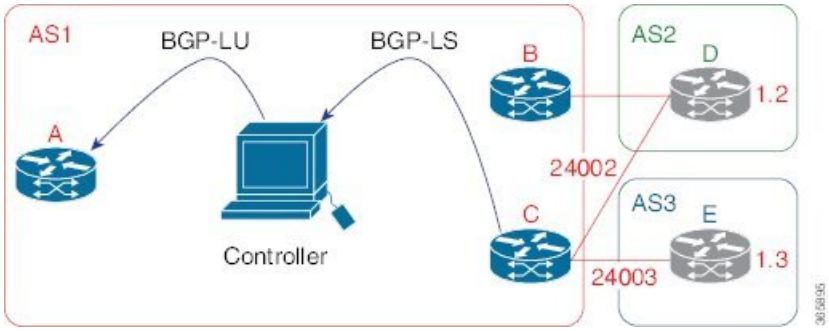
To distribute OSPFv2 link-state data using BGP-LS, use the **distribute link-state** command in router configuration mode.

```
Router# configure
Router(config)# router ospf 100
Router(config-ospf)# distribute link-state instance-id 32
```

# Use Case: 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 10: Topology*




## Procedure

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

**Example:**

```
RP/0/RP0/CPU0:router_C(config)# router bgp 1
RP/0/RP0/CPU0:router_C(config-bgp)# neighbor 192.168.1.3
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# remote-as 3
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# description to E
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# egress-engineering
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_in in
RP/0/RP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_out out
RP/0/RP0/CPU0:router_C(config-bgp-nbr-af)# exit
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# exit
RP/0/RP0/CPU0:router_C(config-bgp)# neighbor 192.168.1.2
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# remote-as 2
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# description to D
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# egress-engineering
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_in in
RP/0/RP0/CPU0:router_C(config-bgp-nbr-af)# route-policy bgp_out out
RP/0/RP0/CPU0:router_C(config-bgp-nbr-af)# exit
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# exit
```

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

**Example:**

```
RP/0/RP0/CPU0:router_C(config-bgp)# neighbor 172.29.50.71
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# remote-as 1
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# description to EPE_controller
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# address-family link-state link-state
RP/0/RP0/CPU0:router_C(config-bgp-nbr)# exit
RP/0/RP0/CPU0:router_C(config-bgp)# exit
```

**Step 3** Configure MPLS static on the egress interfaces connecting to the eBGP peer.

**Example:**

```
RP/0/RP0/CPU0:router_C(config)# mpls static
RP/0/RP0/CPU0:router_C(config-mpls-static)# interface TenGigE 0/3/0/0
```

```
RP/0/RP0/CPU0:router_C(config-mpls-static)# interface TenGigE 0/1/0/0
RP/0/RP0/CPU0:router_C(config-mpls-static)# exit
```

**Step 4** Commit the configuration.

**Example:**

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

**Step 5** Verify the configuration.

**Example:**

```
RP/0/RP0/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: 10.1.1.3
  Remote RID: 10.1.1.4
   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: 10.1.1.3
  Remote RID: 10.1.1.5
   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/RP0/CPU0:router_C# show mpls forwarding labels 24002 24003
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
24002  Pop         No ID              Te0/0/0/1    192.168.1.2     0
24003  Pop         No ID              Te0/0/0/2    192.168.1.3     0
```

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.

```
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)# 10.1.1.1/32 10 range 200
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)# 192.168.64.1/32 400 range 300
```

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

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

### Verification

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

```
RP/0/RSP0/CPU0:router# show segment-routing mapping-server prefix-sid-map ipv4 detail
Prefix
10.1.1.1/32
    SID Index:      10
    Range:          200
    Last Prefix:    10.1.1.200/32
    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 00:11:45Paths: (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 192.168.64.1/32
  Known via "bgp 65000", distance 200, metric 0, [ei]-bgp, labeled SR, type internal
  Installed Oct 25 01:02:28.583 for 00:20:09
  Routing Descriptor Blocks
    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
0x71c42b40) [1], 0x0 (0x71c11590), 0x808 (0x722b91e0)
 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 (0x722eb450) ext 0x0 (0x0)]
  LW-LDI[type=5, refc=3, ptr=0x71c11590, sh-ldi=0x722eb450]
  gateway array update type-time 3 Oct 31 23:49:11.720
 LDI Update time Oct 31 23:23:48.733
 LW-LDI-TS Oct 31 23:23:48.733
   via 20.0.101.1/32, 0 dependencies, recursive, bgp-ext [flags 0x6020]
    path-idx 0 NHID 0x0 [0x7129a294 0x0]
    recursion-via-/32
    unresolved
     local label 16400
     labels imposed {ExpNullv6}


RP/0/RSP0/CPU0:router# show bgp labels
BGP router identifier 2.1.1.1, local AS number 65000
BGP generic scan interval 60 secs
```

```
Non-stop routing is enabled
BGP table state: Active
Table ID: 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
   Network            Next Hop        Rcvd Label      Local Label
*>i10.1.1.1/32        10.1.1.1        3               16010
*> 2.1.1.1/32         0.0.0.0         nolabel         3
*> 192.68.64.1/32     20.0.101.1      2               16400
*> 192.68.64.2/32     20.0.101.1      2               16401
```

# BGP-LU Inter-AS Option-C Interworking with LDP and IGP SR-MPLS using Proxy BGP-SR

***Table 15: Feature History Table***

| Feature Name | Release | Description |
|---|---|---|
| BGP-LU Inter-AS Option-C Interworking with LDP and IGP SR-MPLS using Proxy BGP-SR | Release 7.3.2 | This feature extends the current Proxy BGP-SR functionality by allowing the BGP-LU ASBR router with Proxy BGP-SR configured to also interconnect attached LDP domains.<br>The Proxy BGP-SR feature allows interconnection of IGP SR-MPLS domains and legacy domains via BGP-LU Inter-AS option-C. It provides a prefix-to-SID mapping for BGP-LU prefixes that are learned without a Prefix-SID. |

The Proxy BGP-SR feature allows interconnection of IGP SR-MPLS domains and legacy domains via BGP-LU Inter-AS option-C. It provides a prefix-to-SID mapping for BGP-LU prefixes that are learned without a Prefix-SID. This new feature extends the current functionality by allowing the BGP-LU ASBR router (configured with Proxy BGP-SR) to also interconnect attached LDP domains.

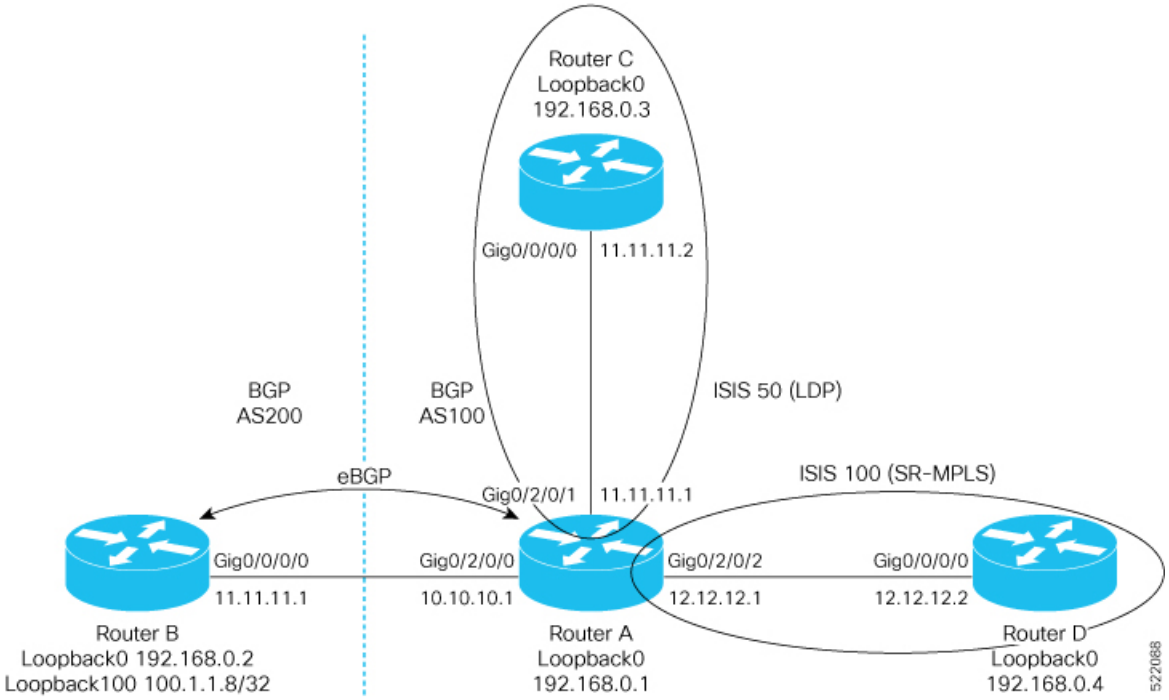
With this enhancement, when performing redistribution from BGP into IGP, LDP would use the same local label assigned by BGP for a prefix learned by BGP-LU. The local label value is based on the SR mapping server configuration (Proxy-BGP SR feature). This behavior allows incoming LDP traffic destined to a redistributed prefix to be switched over to the BGP-LU Inter-AS LSP.

### Use Case

In the following figure, Router A does the following:

- Is an ASBR for BGP AS 100 running BGP-LU with BGP AS 200

- Interconnects two IS-IS processes: one running LDP and another running Segment Routing
- Redistributes prefixes learned by BGP-LU from AS 200 into both IS-IS instances
- Runs SR Mapping Server (SRMS) in order to assign mappings to prefixes learned by BGP LU from AS 200 without a prefix SID (proxy BGP-SR) and prefixes learned from the LDP domain




### Configuration on Router A - ASBR for AS100

```
prefix-set pfxset-bgplu
  100.1.1.8/32  // The Prefix under test
end-set
!
prefix-set LOOPBACKS
  192.168.0.1,
  192.168.0.2,
  192.168.0.3,
  192.168.0.4,
  192.168.0.8
end-set
!
route-policy Pass
  pass
end-policy
!
route-policy rpl-bgplu
  if destination in pfxset-bgplu then
    pass
  else
    drop
  endif
end-policy
!
route-policy MATCH_LOOPBACKS
```

```
  if destination in LOOPBACKS then
    pass
  else
    drop
  endif
end-policy
!
router static
 address-family ipv4 unicast
  10.10.10.2/32 GigabitEthernet0/2/0/0
 !
!
router isis 50
 is-type level-2-only
 net 49.0001.0000.0000.0001.00
 address-family ipv4 unicast
  metric-style wide
  redistribute bgp 100 route-policy rpl-bgplu  // Redistribute prefixes learned by BGP-LU
into IS-IS LDP domain
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/2/0/1
  address-family ipv4 unicast
  !
 !
!
router isis 100
 is-type level-2-only
 net 49.0001.0000.0000.0011.00
 distribute link-state
 address-family ipv4 unicast
  metric-style wide
  mpls traffic-eng level-2-only
  mpls traffic-eng router-id Loopback0
  redistribute bgp 100 route-policy rpl-bgplu  // Redistribute prefixes learned by BGP-LU
into IS-IS SR domain
  segment-routing mpls
  segment-routing prefix-sid-map advertise-local
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid index 1
  !
 !
 interface GigabitEthernet0/2/0/2
  address-family ipv4 unicast
  !
 !
!
router bgp 100
 bgp router-id 192.168.0.1
 address-family ipv4 unicast
  segment-routing prefix-sid-map  // SR Proxy SID Configuration
  network 192.168.0.1/32
  redistribute isis 50 route-policy MATCH_LOOPBACKS
  redistribute isis 100 route-policy MATCH_LOOPBACKS
  allocate-label all
 !
 neighbor 10.10.10.2
```

```
  remote-as 200
  address-family ipv4 labeled-unicast
   route-policy Pass in
   route-policy Pass out
  !
 !
!
mpls ldp
 router-id 192.168.0.1
 interface GigabitEthernet0/2/0/1
 !
!
segment-routing
 global-block 16000 23999
 mapping-server
  prefix-sid-map  // SRMS configuration
   address-family ipv4
    100.1.1.8/32 108 range 1  // SRMS mapping - LU prefix 100.1.1.8/32 assigned prefix index
 108
    192.168.0.3/32 3 range 1  // SRMS mapping - LDP prefix Router C assigned prefix index
3
   !
  !
 !
!
```

### Configuration on Router B - ASBR for AS200

```
route-policy Pass
  pass
end-policy
!
router static
 address-family ipv4 unicast
  10.10.10.1/32 GigabitEthernet0/0/0/0
 !
!
router bgp 200
 bgp router-id 192.168.0.2
 address-family ipv4 unicast
  network 100.1.1.8/32  // Import/Inject route into BGP
  network 192.168.0.2/32
  allocate-label all
 !
 neighbor 10.10.10.1
  remote-as 100
  address-family ipv4 labeled-unicast
   route-policy Pass in
   route-policy Pass out
  !
 !
!
```

### Configuration on Router C in the LDP Domain

```
router isis 50
 is-type level-2-only
 net 49.0001.0000.0000.0003.00
 address-family ipv4 unicast
  metric-style wide
 !
 interface Loopback0
  passive
```

```
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/0/0/0
  address-family ipv4 unicast
  !
 !
!
mpls ldp
 router-id 192.168.0.3
 interface GigabitEthernet0/0/0/0
 !
!
```

### Configuration on Router D in the SR IS-IS Domain

```
router isis 100
 is-type level-2-only
 net 49.0001.0000.0000.0004.00
 address-family ipv4 unicast
  metric-style wide
  mpls traffic-eng level-2-only
  mpls traffic-eng router-id Loopback0
  segment-routing mpls
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid index 4
  !
 !
 interface GigabitEthernet0/0/0/0
  address-family ipv4 unicast
  !
 !
!
segment-routing
!
```

# Optimal Utilization of ECMP FEC Resources

*Table 16: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Optimal Utilization of ECMP FEC Resources | Release 7.5.2 | BGP-SR multipath ECMP FEC optimization is enhanced to support 32k BGP-LU prefixes (from the earlier 4k BGP-LU prefixes) on multipath with the same outgoing label. This results in the consumption of lesser ECMP FEC resources, thus avoiding out-of-resource (OOR) situations for your router.<br><br>In earlier releases, all 4k BGP-LU prefixes consumed all the 4k ECMP FEC resources.<br><br>Use the **hw-module fib mpls bgp-sr lsr-optimized** command to enable BGP-SR multipath ECMP FEC optimization. |

The BGP-SR multipath ECMP FEC optimization solution minimizes the ECMP FEC resource consumption during underlay programming for an SR-MPLS network. BGP-LU prefix consumes one FEC resource for every path and one ECMP FEC resource for multipath. When you configure BGP-LU multipath, each BGP-LU prefix consumes one ECMP FEC resource for programming the prefix. This limits the BGP-LU prefix scale to only 4k. To support higher BGP-LU prefix scales of upto 32k, you need to conserve the ECMP FEC resources. With BGP-SR multipath ECMP FEC optimization feature, you can conserve the ECMP FEC resource usage when BGP-LU multipath is configured.

Enable the **hw-module fib mpls bgp-sr lsr-optimized** command, and ensure that all BGP-LU prefix paths advertise the same `out_label`. You can achieve this with BGP-SR or proxy BGP-SR by using same **prefix-sid-map** on the next hop routers.

After you enable ECMP FEC optimization, all BGP-LU prefix is assigned the same ECMP FEC key by conserving the ECMP FEC resources and supports scale of upto 32k BGP-LU prefixes.

**Note** If the paths for BGP-LU prefixes don't have the same `out_label`, then each prefix whose `out_label` isn't the same, starts to consume ECMP FEC resources and may result in out-of-resource (OOR) when it exceeds 4k, and you may observe traffic drops. Also, the successive prefixes starts to consume FEC resources, which affect multipath support.

### Usage Guidelines and Limitations

- BGP-SR multipath ECMP FEC optimization feature isn't supported on Cisco NCS 5700 series fixed port routers or Cisco NCS 5500 series routers that have the Cisco NC57 line cards installed and operating in the native or compatible modes.
- All the prefixes advertised must be /32 (IPv4 only) and to enable optimization, all prefixes must have the same outgoing label.
- eBGP is always interface peering and iBGP is always loopback peering.
- Supports 32k LU prefix scale (IPv4 only) for loopback peering and 24k LU prefix scale for interface peering.
  - For eBGP interface peering, the maximum BGP next hops possible is only 2.
  - For iBGP loopback peering, the maximum BGP next hops possible is only 4.
- You can't configure the **hw-module fib mpls bgp-sr lsr-optimized** command, if **hw-module fib mpls label lsr-optimized** command is already configured.
- No ECMP FEC optimization is supported for L3VPN services over BGP-LU loopback peering.

### Enable BGP-SR Multipath ECMP FEC Optimization

To enable BGP-SR multipath ECMP FEC optimization, you must configure the **hw-module fib mpls label lsr-optimized** command in global configuration mode. After enabling this feature, reload the chassis.

```
Router(config)#hw-module fib mpls bgp-sr lsr-optimized
Tue Nov 16 22:27:42.360 UTC
In order to activate this MPLS profile, you must manually reload the chassis/all line cards
Router(config)#commit

Router# reload location 0/0/CPU0

Proceed with reload? [confirm]
Reloading node 0/0/CPU0
```

### Verification

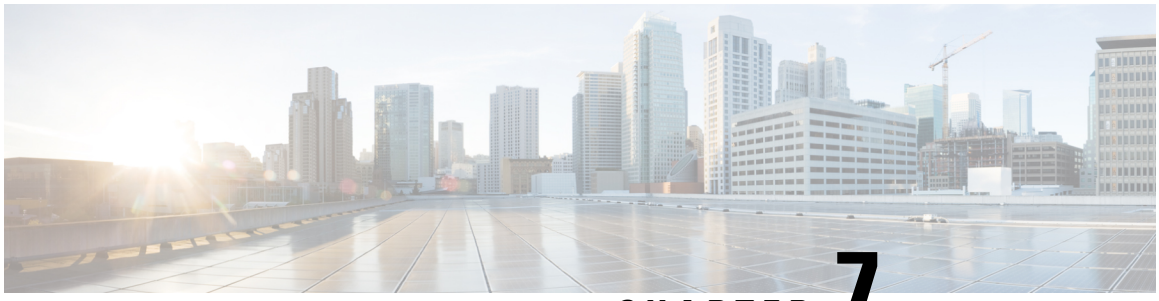
The following example shows NPU ECMP FEC resource before enabling BGP-SR multipath ECMP FEC optimization, shows the OOR state and the ECMP FEC resource consumption.

```
RP/0/RP0/CPU0:PE1#show controllers npu resources ecmpfec location all
Tue Nov 16 21:43:01.219 UTC
HW Resource Information For Location: 0/7/CPU0
HW Resource Information
    Name                            : ecmp_fec
    Asic Type                       : Jericho
NPU-0
OOR Summary
        Estimated Max Entries       : 4096
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                   : Red
        OOR State Change Time       : 2021.Nov.16 21:39:21 UTC
        Bank Info                   : ECMP
```

```
OFA Table Information
(May not match HW usage)
        ipnhgroup                   : 3916
        ip6nhgroup                  : 178

Current Hardware Usage
    Name: ecmp_fec
        Estimated Max Entries       : 4096
        Total In-Use              : 4094     (99 %)
        OOR State                   : Red
        OOR State Change Time       : 2021.Nov.16 21:39:21 UTC
        Bank Info                   : ECMP
       Name: hier_0
           Total In-Use               : 4094
           OOR State                  : Red
           OOR State Change Time      : 2021.Nov.16 21:39:21 UTC
           Bank Info                  : ECMP
```

The following example shows NPU usage after enabling BGP-SR multipath ECMP FEC optimization, shows improvement in the OOR state and the ECMP FEC resource consumption.

```
RP/0/RP0/CPU0:PE1#show controllers npu resources ecmpfec location all
Wed Nov 17 19:49:08.978 UTC
HW Resource Information
    Name                            : ecmp_fec
    Asic Type                       : Jericho
NPU-0
OOR Summary
        Estimated Max Entries       : 4096
        Red Threshold               : 95 %
        Yellow Threshold            : 80 %
        OOR State                 : Green
        Bank Info                   : ECMP
OFA Table Information
(May not match HW usage)
        ipnhgroup                   : 185
        ip6nhgroup                  : 178
Current Hardware Usage
    Name: ecmp_fec
        Estimated Max Entries       : 4096
        Total In-Use                : 363      (8 %)
        OOR State                   : Green
        Bank Info                   : ECMP
       Name: hier_0
           Total In-Use               : 363
           OOR State                  : Green
           Bank Info                  : ECMP
```



CHAPTER 7

# Configure SR-TE Policies

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



## SR-TE Policy Overview

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. See *SR-TE Policy Path Types* section for more information.

# Usage Guidelines and Limitations

*Table 17: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| L3VPN BGP PIC over SR-TE | Release 7.3.2 | This feature provides BGP PIC support for L3VPN over SR policies. BGP PIC provides fast convergence when traffic switches from a primary path to a backup path.<br>BGP PIC over SR-TE is supported when both primary and backup paths each resolve into the BSID of an SR policy. |

Observe the following guidelines and limitations for the platform.

- Broadcast links are not supported, configure IGP's interface as P2P (point-to-point).
- The ECMP path-set of an IGP route with a mix of SR-TE Policy paths (Autoroute Include) and unprotected native paths is supported.
- The ECMP path-set of an IGP route with a mix of SR-TE Policy paths (Autoroute Include) and protected (LFA/TI-LFA) native paths is not supported.
- Before configuring SR-TE policies, use the **distribute link-state** command under IS-IS or OSPF to distribute the link-state database to external services.
- L3VPN BGP PIC over SR-TE is supported.

  BGP PIC over SR-TE is supported when both primary and backup paths each resolve into the BSID of an SR policy. BGP PIC over SR-TE is not supported when primary and backup paths are of different resolution types. For example, when a primary path resolves into the BSID of an SR policy, the backup path cannot point to a native LSP. When this happens, the backup path will not be programmed. For information about BGP PIC, refer to the BGP PIC chapter in the *BGP Configuration Guide for Cisco NCS 540 Series Routers* .
- SR-TE over BVI is not supported. An SR-TE policy cannot be resolved over an MPLS-enabled BVI interface.
- Counter implications when BVI and SR-TE co-exist in same NPU—Counters for a BVI's logical interface are not allocated when the same NPU hosts layer-2 (sub)interface(s) associated with the BVI alongside other port(s) used as egress interface(s) for an SR policy
- GRE tunnel as primary interface for an SR policy is not supported.
- GRE tunnel as backup interface for an SR policy with TI-LFA protection is not supported.
- Head-end computed inter-domain SR policy with Flex Algo constraint and IGP redistribution is not supported. This is supported with Flex Algo-aware path computation at SR-PCE, with or without IGP redistribution. See SR-PCE Flexible Algorithm Multi-Domain Path Computation, on page 304.

# Instantiation of an SR Policy

An SR policy is instantiated, or implemented, at the head-end router.

The following sections provide details on the SR policy instantiation methods:

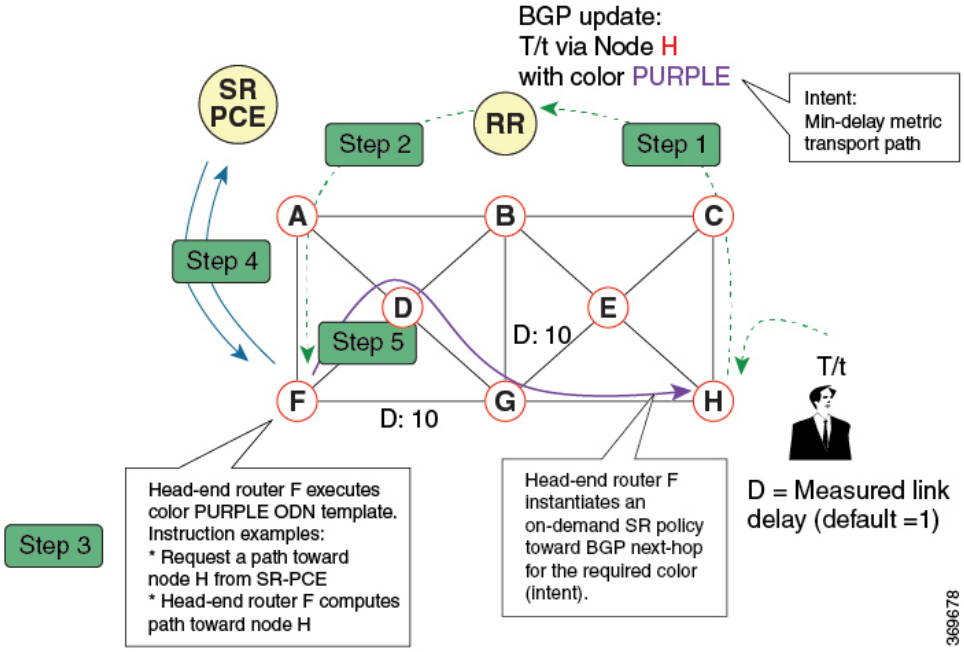
- On-Demand SR Policy – SR On-Demand Next-Hop , on page 165
- Manually Provisioned SR Policy, on page 198
- PCE-Initiated SR Policy, on page 198

## 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:

- EVPN-VPWS (single-homing)
- EVPN-VPWS (multi-homing)
- EVPN (single-homing/multi-homing)

**Note** For EVPN single-homing, you must configure an EVPN Ethernet Segment Identifier (ESI) with a non-zero value.

**Note** Colored per-ESI/per-EVI EVPN Ethernet Auto-Discovery route (route-type 1) and Inclusive Multicast Route (route-type 3) are used to trigger instantiation of ODN SR-TE policies.

**Note** The following scenarios involving virtual Ethernet Segments (vES) are also supported with EVPN ODN:

- VPLS VFI as vES for single-active Multi-Homing to EVPN
- Active/backup Pseudo-wire (PW) as vES for Single-Homing to EVPN
- Static Pseudo-wire (PW) as vES for active-active Multi-Homing to EVPN

## SR-ODN 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 computation:

   a. Configure SR-PCE. For detailed SR-PCE configuration information, see Configure SR-PCE, on page 298.

   b. 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.

2. Define BGP color extended communities. Refer to the "Implementing BGP" chapter in the *BGP Configuration Guide for Cisco NCS 540 Series Routers*.

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 NCS 540 Series Routers*.

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

**Note** The following table shows the supported RPL match operations; however, routing policies are required primarily to set BGP color extended community. Matching based on BGP color extended communities is performed automatically by ODN's on-demand color template.

| Attach Point | Set | Match |
|---|---|---|
| VRF export | X | X |
| VRF import | – | X |
| Neighbor-in | X | X |
| Neighbor-out | X | X |
| Inter-AFI export | – | X |
| Inter-AFI import | – | X |
| Default-originate | X | – |

4. Apply routing policies to a service. Refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco NCS 540 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 10
```

**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 10 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 10 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 te
```

- 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 5
```

### 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 775 type link
```

- Use the **affinity** {**include-any** | **include-all** | **exclude-any**} {**name** *WORD*} command to configure the affinity constraints.

```
Router(config-sr-te-color-dyn)# affinity exclude-any name CROSS
```

- 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 (12).

```
Router(config-sr-te-color)# maximum-sid-depth 5
```

  See Customize MSD Value at PCC, on page 218 for information about SR-TE label imposition capabilities.

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

```
Router(config-sr-te-color-dyn)# sid-algorithm 128
```

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

```
Router(config-sr-te-color)# constraints segments sid-algorithm 128
```

- Use the **constraints segments protection** {**protected-only** | **protected-preferred** | **unprotected-only** | **unprotected-preferred**} command to configure the Adj-SID protection behavior constraints.

```
Router(config-sr-te-color)# constraints segments protection protected-only
```

See Segment Protection-Type Constraint, on page 207 for information about Adj-SID protection behavior.

# Configuring SR-ODN: Examples

## Configuring SR-ODN: Layer-3 Services 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 22: minimization objective = te-metric; path computation at SR-PCE; constraints = affinity

```
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 22
   dynamic
    pcep
    !
    metric
     type te
    !
    affinity exclude-any
     name CROSS
    !
   !
  !
  !
  on-demand color 128
   constraints
    segments
     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 examples include the set color action only. The last RPL example performs the set color action for selected destinations based on a prefix-set.

```
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
!

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.

```
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
 !
!

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

### 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 10.1.1.8.

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

BGP VRF vrf_cust1, state: Active
BGP Route Distinguisher: 10.1.1.4:101
VRF ID: 0x60000007
BGP router identifier 10.1.1.4, local AS number 100
Non-stop routing is enabled
```

```
BGP table state: Active
Table ID: 0xe0000007   RD version: 282
BGP main routing table version 287
BGP NSR Initial initsync version 31 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0

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: 10.1.1.4:101 (default for vrf vrf_cust1)
*> 44.1.1.0/24        40.4.101.11                            0 400 {1} i
*>i55.1.1.0/24        10.1.1.5                     100       0 500 {1} i
*>i88.1.1.0/24        10.1.1.8 C:10                100       0 800 {1} i
*>i99.1.1.0/24        10.1.1.9                     100       0 800 {1} i

Processed 4 prefixes, 4 paths
```

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: 10.1.1.4:101
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}
    10.1.1.8 C:10 (bsid:24036) (metric 20) from 10.1.1.55 (10.1.1.8)
      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 RT:100:1
      Originator: 10.1.1.8, Cluster list: 10.1.1.55
      SR policy color 10, up, registered, bsid 24036, if-handle 0x08000024

     Source AFI: VPNv4 Unicast, Source VRF: default, Source Route Distinguisher: 10.1.1.8:101
```

### 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

Prefix              Next Hop            Interface
------------------- ------------------- ------------------
0.0.0.0/0           drop                default handler
0.0.0.0/32          broadcast
40.4.101.0/24       attached            TenGigE0/0/0/0.101
40.4.101.0/32       broadcast           TenGigE0/0/0/0.101
40.4.101.4/32       receive             TenGigE0/0/0/0.101
40.4.101.11/32      40.4.101.11/32      TenGigE0/0/0/0.101
```

```
40.4.101.255/32     broadcast           TenGigE0/0/0/0.101
44.1.1.0/24         40.4.101.11/32      <recursive>
55.1.1.0/24         10.1.1.5/32         <recursive>
88.1.1.0/24         24036 (via-label)   <recursive>
99.1.1.0/24         10.1.1.9/32         <recursive>
224.0.0.0/4         0.0.0.0/32
224.0.0.0/24        receive
255.255.255.255/32  broadcast
```

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.

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

88.1.1.0/24, version 51, internal 0x5000001 0x0 (ptr 0x98c60ddc) [1], 0x0 (0x0), 0x208
(0x98425268)
 Updated May 20 09:23:34.216
 Prefix Len 24, traffic index 0, precedence n/a, priority 3
   via local-label 24036, 5 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x97091ec0 0x0]
    recursion-via-label
    next hop VRF - 'default', table - 0xe0000000
    next hop via 24036/0/21
     next hop srte_c_10_ep labels imposed {ImplNull 24012}
```

### 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 10.1.1.8.

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

 Color             Endpoint  Admin   Oper              Binding
                             State  State                 SID
------ -------------------- ------ ------ --------------------
    10              10.1.1.8    up     up                24036
```

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

**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.

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

SR-TE policy database
---------------------

Color: 10, End-point: 10.1.1.8
  Name: srte_c_10_ep_10.1.1.8
  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_10.1.1.8_discr_200
        PLSP-ID: 12
      Dynamic (valid)
        Metric Type: TE,   Path Accumulated Metric: 30
            16009 [Prefix-SID, 10.1.1.9]
            16008 [Prefix-SID, 10.1.1.8]
    Preference: 100 (BGP ODN)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_10_ep_10.1.1.8_discr_100
        PLSP-ID: 11
      Dynamic (pce 10.1.1.57) (valid)
        Metric Type: TE,   Path Accumulated Metric: 30
            16009 [Prefix-SID, 10.1.1.9]
            16008 [Prefix-SID, 10.1.1.8]
  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 10.1.1.8.

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

Color Endpoint        Segment      Outgoing Outgoing     Next Hop     Bytes        Pure
                      List         Label    Interface                Switched     Backup
----- --------------- ------------ -------- ------------ ------------ ------------ ------
10    10.1.1.8        dynamic      16009    Gi0/0/0/4    10.4.5.5     0
                                   16001    Gi0/0/0/5    11.4.8.8     0            Yes

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: 10.1.1.8
  Name: srte_c_10_ep_10.1.1.8
  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: EVPN Services Examples

### 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.

```
extcommunity-set opaque color-44
  44
end-set

extcommunity-set opaque color-55
  55
end-set

extcommunity-set opaque color-77
  77
end-set

extcommunity-set opaque color-88
  88
end-set
```

### Configuring RPL to Set BGP Color (EVPN Services): Examples

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

The following RPL examples match on EVPN route-types and then set the BGP color extended community.

```
route-policy sample-export-rpl
  if evpn-route-type is 1 then
    set extcommunity color color-44
  endif
  if evpn-route-type is 3 then
    set extcommunity color color-55
  endif
end-policy

route-policy sample-import-rpl
  if evpn-route-type is 1 then
    set extcommunity color color-77
  elseif evpn-route-type is 3 then
    set extcommunity color color-88
  else
    pass
  endif
end-policy
```

The following RPL example sets BGP color extended community while matching on the following:

- Route Distinguisher (RD)
- Ethernet Segment Identifier (ESI)

- Ethernet Tag (ETAG)
- EVPN route-types

```
route-policy sample-bgpneighbor-rpl
  if rd in (10.1.1.1:3504) then
    set extcommunity color color3504
  elseif rd in (10.1.1.1:3505) then
    set extcommunity color color3505
  elseif rd in (10.1.1.1:3506) then
    set extcommunity color color99996
  elseif esi in (0010.0000.0000.0000.1201) and rd in (10.1.1.1:3508) then
    set extcommunity color color3508
  elseif etag in (30509) and rd in (10.1.1.1:3509) then
    set extcommunity color color3509
  elseif etag in (0) and rd in (10.1.1.1:2001) and evpn-route-type is 1 then
    set extcommunity color color82001
  elseif etag in (0) and rd in (10.1.1.1:2001) and evpn-route-type is 3 then
    set extcommunity color color92001
endif
  pass
end-policy
```

### Applying RPL to BGP Services (EVPN Services): Example

The following examples show various RPLs that set BGP color community being applied to EVPN services.

The following 2 examples show the RPL applied at the EVI export and import attach-points.

**Note** RPLs applied under EVI import or export attach-point also support matching on the following:

- Ethernet Segment Identifier (ESI)
- Ethernet Tag (ETAG)
- EVPN-Originator

```
evpn
 evi 101
  bgp
   route-target 101:1
   route-target import 100:1
   route-target export 101:1
   route-policy import sample-import-rpl
  !
  advertise-mac
  !
 !
 evi 102
  bgp
   route-target 102:1
   route-target import 100:2
   route-target export 102:1
   route-policy export sample-export-rpl
  !
  advertise-mac
  !
 !
!
```

The following example shows the RPL applied at the BGP neighbor-out attach-point.

**Note** RPLs defined under BGP neighbor-out attach-point also support matching on the following:

- EVPN-Originator

```
router bgp 100
 bgp router-id 10.1.1.1
 address-family l2vpn evpn
 !
 neighbor-group evpn-rr
  remote-as 100
  update-source Loopback0
  address-family l2vpn evpn
  !
  neighbor 10.10.10.10
   use neighbor-group evpn-rr
   address-family l2vpn evpn
    route-policy sample-bgpneighbor-rpl out
```

## 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 11: Topology for Use Case: SR-ODN for EVPN-VPWS***

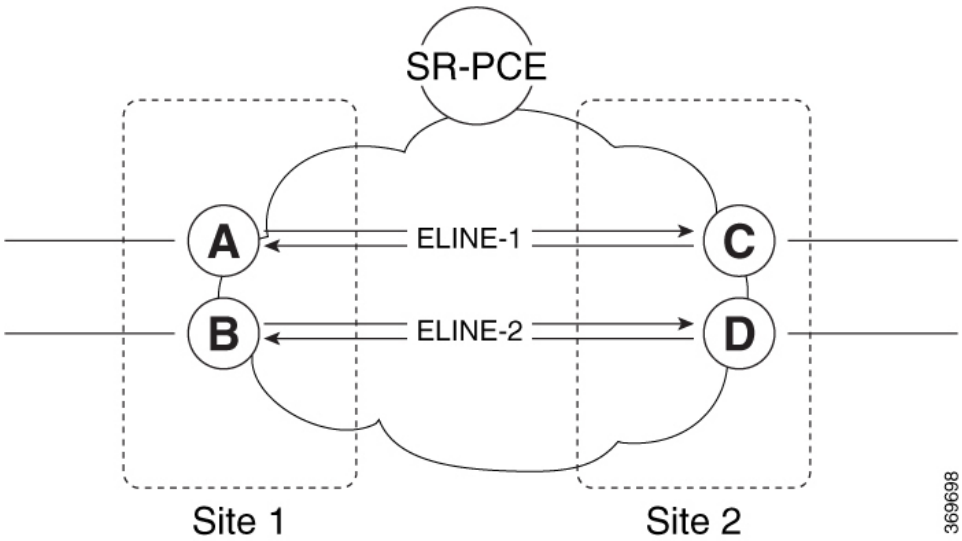


*Table 18: Use Case Parameters*

| | | |
|---|---|---|
| **IP Addresses of Loopback0 (Lo0) Interfaces** | SR-PCE Lo0: 10.1.1.207 | |
| | Site 1:<br>• Node A Lo0: 10.1.1.5<br>• Node B Lo0: 10.1.1.6 | Site 2:<br>• Node C Lo0: 10.1.1.2<br>• Node D Lo0: 10.1.1.4 |
| **EVPN-VPWS Service Parameters** | ELINE-1:<br>• EVPN-VPWS EVI 100<br>• Node A: AC-ID = 11<br>• Node C: AC-ID = 21 | ELINE-2:<br>• EVPN-VPWS EVI 101<br>• Node B: AC-ID = 12<br>• Node D: AC-ID = 22 |
| **ODN BGP Color Extended Communities** | Site 1 routers (Nodes A and B):<br>• set color 10000<br>• match color 11000 | Site 2 routers (Nodes C and D):<br>• set color 11000<br>• match color 10000 |
| **Note**<br>These colors are associated with the EVPN route-type 1 routes of the EVPN-VPWS services. | | |
| **PCEP LSP Disjoint-Path Association Group ID** | Site 1 to Site 2 LSPs (from Node A to Node C/from Node B to Node D):<br>• group-id = 775 | Site 2 to Site 1 LSPs (from Node C to Node A/from Node D to Node B):<br>• group-id = 776 |

The use case provides configuration and verification outputs for all devices.

| Configuration | Verification |
|---|---|
| Configuration: SR-PCE, on page 178 | Verification: SR-PCE, on page 183 |
| Configuration: Site 1 Node A, on page 179 | Verification: Site 1 Node A, on page 187 |
| Configuration: Site 1 Node B, on page 180 | Verification: Site 1 Node B, on page 189 |
| Configuration: Site 2 Node C, on page 181 | Verification: Site 2 Node C, on page 192 |
| Configuration: Site 2 Node D, on page 182 | Verification: Site 2 Node D, on page 195 |

### 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 301 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.

```
/* 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 */

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 '.*..*..*..*:(100)') then
    set extcommunity color color-10000
  endif
  pass
end-policy
!
router bgp 65000
 neighbor 10.1.1.253
  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 1 Node B

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

```
/* 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 10.1.1.253
  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 10.1.1.253
  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 10.1.1.253
  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/CPU0:SR-PCE# show pce ipv4 peer
Mon Jul 15 19:41:43.622 UTC

PCE's peer database:
--------------------
Peer address: 10.1.1.2
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 10.1.1.4
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 10.1.1.5
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 10.1.1.6
  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 (10.1.1.5): LSP symbolic name = bgp_c_11000_ep_10.1.1.2_discr_100
- At Node B (10.1.1.6): LSP symbolic name = bgp_c_11000_ep_10.1.1.4_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 10.1.1.6, tunnel name bgp_c_11000_ep_10.1.1.4_discr_100,  PLSP ID 18, tunnel ID 17,
 LSP ID 3, Configured on PCC
  LSP[1]:
   PCC 10.1.1.5, tunnel name bgp_c_11000_ep_10.1.1.2_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 (10.1.1.5) that is used to carry traffic of EVPN VPWS EVI 100 towards node C (10.1.1.2).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.5 name bgp_c_11000_ep_10.1.1.2_discr_100
Thu Jul 11 03:58:45.903 UTC

PCE's tunnel database:
```

```
----------------------
PCC 10.1.1.5:

Tunnel Name: bgp_c_11000_ep_10.1.1.2_discr_100
Color: 11000
Interface Name: srte_c_11000_ep_10.1.1.2
 LSPs:
  LSP[0]:
   source 10.1.1.5, destination 10.1.1.2, 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: 10.1.1.5
   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 10.1.1.7
      SID[2]: Node, Label 16002, Address 10.1.1.2
   Computed path: (Local PCE)
     Computed Time: Thu Jul 11 03:49:48 UTC 2019 (00:08:58 ago)
     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 10.1.1.7
      SID[2]: Node, Label 16002, Address 10.1.1.2
   Recorded path:
     None
   Disjoint Group Information:
     Type Link-Disjoint, Group 775
```

This output shows details for the LSP at Node B (10.1.1.6) that is used to carry traffic of EVPN VPWS EVI 101 towards node D (10.1.1.4).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.6 name bgp_c_11000_ep_10.1.1.4_discr_100
Thu Jul 11 03:58:56.812 UTC

PCE's tunnel database:
----------------------
PCC 10.1.1.6:

Tunnel Name: bgp_c_11000_ep_10.1.1.4_discr_100
Color: 11000
Interface Name: srte_c_11000_ep_10.1.1.4
 LSPs:
  LSP[0]:
   source 10.1.1.6, destination 10.1.1.4, 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: 10.1.1.6
  LSP is subdelegated to: None
  Reported path:
    Metric type: IGP, Accumulated Metric 40
     SID[0]: Node, Label 16001, Address 10.1.1.1
     SID[1]: Node, Label 16004, Address 10.1.1.4
  Computed path: (Local PCE)
    Computed Time: Thu Jul 11 03:49:48 UTC 2019 (00:09:08 ago)
    Metric type: IGP, Accumulated Metric 40
     SID[0]: Node, Label 16001, Address 10.1.1.1
     SID[1]: Node, Label 16004, Address 10.1.1.4
  Recorded path:
    None
  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 (10.1.1.2): LSP symbolic name = bgp_c_10000_ep_10.1.1.5_discr_100
- At Node D (10.1.1.4): LSP symbolic name = bgp_c_10000_ep_10.1.1.6_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 10.1.1.4, tunnel name bgp_c_10000_ep_10.1.1.6_discr_100,  PLSP ID 16, tunnel ID 14,
 LSP ID 1, Configured on PCC
  LSP[1]:
   PCC 10.1.1.2, tunnel name bgp_c_10000_ep_10.1.1.5_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 (10.1.1.2) that is used to carry traffic of EVPN VPWS EVI 100 towards node A (10.1.1.5).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.2 name bgp_c_10000_ep_10.1.1.5_discr_100
Thu Jul 11 03:55:21.706 UTC

PCE's tunnel database:
----------------------
PCC 10.1.1.2:

Tunnel Name: bgp_c_10000_ep_10.1.1.5_discr_100
Color: 10000
Interface Name: srte_c_10000_ep_10.1.1.5
 LSPs:
  LSP[0]:
   source 10.1.1.2, destination 10.1.1.5, 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: 10.1.1.2
  LSP is subdelegated to: None
  Reported path:
    Metric type: IGP, Accumulated Metric 40
     SID[0]: Node, Label 16007, Address 10.1.1.7
     SID[1]: Node, Label 16008, Address 10.1.1.8
     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 10.1.1.7
     SID[1]: Node, Label 16008, Address 10.1.1.8
     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 (10.1.1.4) used to carry traffic of EVPN VPWS EVI 101 towards node B (10.1.1.6).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.4 name bgp_c_10000_ep_10.1.1.6_discr_100
Thu Jul 11 03:55:23.296 UTC

PCE's tunnel database:
----------------------
PCC 10.1.1.4:

Tunnel Name: bgp_c_10000_ep_10.1.1.6_discr_100
Color: 10000
Interface Name: srte_c_10000_ep_10.1.1.6
 LSPs:
  LSP[0]:
   source 10.1.1.4, destination 10.1.1.6, 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: 10.1.1.4
   LSP is subdelegated to: None
   Reported path:
     Metric type: IGP, Accumulated Metric 40
      SID[0]: Node, Label 16001, Address 10.1.1.1
      SID[1]: Node, Label 16006, Address 10.1.1.6
   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 10.1.1.1
      SID[1]: Node, Label 16006, Address 10.1.1.6
   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 10.1.1.5:100). The output includes an EVPN route-type 1 route with color 11000 originated at Node C (10.1.1.2).

```
RP/0/RSP0/CPU0:Node-A# show bgp l2vpn evpn rd 10.1.1.5:100
Wed Jul 10 18:57:57.704 PST
BGP router identifier 10.1.1.5, 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
              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: 10.1.1.5:100 (default for vrf VPWS:100)
*> [1][0000.0000.0000.0000.0000][11]/120
                      0.0.0.0                                0 i
*>i[1][0000.0000.0000.0000.0000][21]/120
                      10.1.1.2 C:11000              100      0 i
```

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 10.1.1.5:100
[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:
10.1.1.5:100
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
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    10.1.1.2 C:11000 (bsid:80044) (metric 40) from 10.1.1.253 (10.1.1.2)
      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 RT:65000:100
      Originator: 10.1.1.2, Cluster list: 10.1.1.253
      SR policy color 11000, up, registered, bsid 80044, if-handle 0x00001b20

     Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 10.1.1.2:100
```

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
Wed Jul 10 18:58:02.333 PST
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect                   Segment 1                       Segment 2
Group      Name       ST   Description            ST       Description             ST
------------------------   ------------------------------   ------------------------------
evpn_vpws_group
           evpn_vpws_100
                      UP   Gi0/0/0/3.2500         UP       EVPN 100,21,10.1.1.2    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 10.1.1.2 (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 10.1.1.2, PW ID: evi 100, ac-id 21, state is up ( established )
    XC ID 0xa0000007
    Encapsulation MPLS
    Source address 10.1.1.5
    Encap type Ethernet, control word enabled
    Sequencing not set
    Preferred path Active : SR TE srte_c_11000_ep_10.1.1.2, On-Demand, fallback enabled
    Tunnel : Up
    Load Balance Hashing: src-dst-mac

      EVPN         Local                          Remote
      ------------ ------------------------------ ------------------------------
      Label        80040                          80056
      MTU          1500                           1500
      Control word enabled                        enabled
      AC ID        11                             21
      EVPN type    Ethernet                       Ethernet

      ------------ ------------------------------ ------------------------------
    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 (10.1.1.2).

```
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
                             State  State                 SID
------ -------------------- ------ ------ -------------------
 11000              10.1.1.2     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 (10.1.1.207).

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_10.1.1.2) is link-disjoint from LSP at Node B (srte_c_11000_ep_10.1.1.4).

```
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: 10.1.1.2
  Name: srte_c_11000_ep_10.1.1.2
  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_10.1.1.2_discr_200
        PLSP-ID: 19
      Dynamic (invalid)
    Preference: 100 (BGP ODN) (active)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_11000_ep_10.1.1.2_discr_100
        PLSP-ID: 18
      Dynamic (pce 10.1.1.207) (valid)
        Metric Type: IGP,   Path Accumulated Metric: 40
          80003 [Adjacency-SID, 11.5.8.5 - 11.5.8.8]
          16007 [Prefix-SID, 10.1.1.7]
          16002 [Prefix-SID, 10.1.1.2]
  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 10.1.1.6:101). The output includes an EVPN route-type 1 route with color 11000 originated at Node D (10.1.1.4).

```
RP/0/RSP0/CPU0:Node-B# show bgp l2vpn evpn rd 10.1.1.6:101
Wed Jul 10 19:08:54.964 PST
BGP router identifier 10.1.1.6, 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-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network            Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 10.1.1.6:101 (default for vrf VPWS:101)
*> [1][0000.0000.0000.0000.0000][12]/120
                      0.0.0.0                                0 i
*>i[1][0000.0000.0000.0000.0000][22]/120
                      10.1.1.4 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.

```
RP/0/RSP0/CPU0:Node-B# show bgp l2vpn evpn rd 10.1.1.6:101
[1][0000.0000.0000.0000.0000][22]/120
Wed Jul 10 19:08:55.039 PST
BGP routing table entry for [1][0000.0000.0000.0000.0000][22]/120, Route Distinguisher:
10.1.1.6:101
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker               322         322
Last Modified: Jul 10 18:42:10.408 for 00:26:44
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    10.1.1.4 C:11000 (bsid:80061) (metric 40) from 10.1.1.253 (10.1.1.4)
      Received Label 80045
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate,
imported, rib-install
      Received Path ID 0, Local Path ID 1, version 319
      Extended community: Color:11000 RT:65000:101
      Originator: 10.1.1.4, Cluster list: 10.1.1.253
      SR policy color 11000, up, registered, bsid 80061, if-handle 0x00000560

     Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 10.1.1.4:101
```

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

```
RP/0/RSP0/CPU0:Node-B# show l2vpn xconnect group evpn_vpws_group
Wed Jul 10 19:08:56.388 PST
```

```
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect                    Segment 1                       Segment 2
Group      Name       ST    Description            ST       Description            ST
-------------------------   ------------------------------  ------------------------------
evpn_vpws_group
           evpn_vpws_101
                      UP    Te0/3/0/0/8.2500       UP       EVPN 101,22,10.1.1.4   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 10.1.1.4 (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 10.1.1.4, PW ID: evi 101, ac-id 22, state is up ( established )
    XC ID 0xa0000009
    Encapsulation MPLS
    Source address 10.1.1.6
    Encap type Ethernet, control word enabled
    Sequencing not set
    Preferred path Active : SR TE srte_c_11000_ep_10.1.1.4, On-Demand, fallback enabled
    Tunnel : Up
    Load Balance Hashing: src-dst-mac

      EVPN         Local                          Remote
      ------------ ------------------------------ ------------------------------
      Label        80060                          80045
      MTU          1500                           1500
      Control word enabled                        enabled
      AC ID        12                             22
      EVPN type    Ethernet                       Ethernet

      ------------ ------------------------------ ------------------------------
    Create time: 10/07/2019 18:32:49 (00:36:06 ago)
    Last time status changed: 10/07/2019 18:42:07 (00:26:49 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 80061 that was triggered by EVPN RT1 prefix with color 11000 advertised by node D (10.1.1.4).

```
RP/0/RSP0/CPU0:Node-B# show segment-routing traffic-eng policy color 11000 tabular
Wed Jul 10 19:08:56.146 PST

 Color             Endpoint  Admin   Oper              Binding
```

```
                            State  State                 SID
------ -------------------- ------ ------ --------------------
 11000              10.1.1.4     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 (10.1.1.207).

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 B (srte_c_11000_ep_10.1.1.4) is link-disjoint from LSP at Node A (srte_c_11000_ep_10.1.1.2).

```
RP/0/RSP0/CPU0:Node-B# show segment-routing traffic-eng policy color 11000
Wed Jul 10 19:08:56.207 PST

SR-TE policy database
---------------------

Color: 11000, End-point: 10.1.1.4
  Name: srte_c_11000_ep_10.1.1.4
  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_10.1.1.4_discr_200
        PLSP-ID: 19
      Dynamic (invalid)
    Preference: 100 (BGP ODN) (active)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_11000_ep_10.1.1.4_discr_100
        PLSP-ID: 18
      Dynamic (pce 10.1.1.207) (valid)
        Metric Type: IGP,   Path Accumulated Metric: 40
          16001 [Prefix-SID, 10.1.1.1]
          16004 [Prefix-SID, 10.1.1.4]
  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 10.1.1.2:100). The output includes an EVPN route-type 1 route with color 10000 originated at Node A (10.1.1.5).

```
RP/0/RSP0/CPU0:Node-C# show bgp l2vpn evpn rd 10.1.1.2:100
BGP router identifier 10.1.1.2, 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
              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: 10.1.1.2:100 (default for vrf VPWS:100)
*>i[1][0000.0000.0000.0000.0000][11]/120
                      10.1.1.5 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 10.1.1.2:100
[1][0000.0000.0000.0000.0000][11]/120
BGP routing table entry for [1][0000.0000.0000.0000.0000][11]/120, Route Distinguisher:
10.1.1.2:100
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
    10.1.1.5 C:10000 (bsid:80058) (metric 40) from 10.1.1.253 (10.1.1.5)
      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: 10.1.1.5, Cluster list: 10.1.1.253
      SR policy color 10000, up, registered, bsid 80058, if-handle 0x000006a0

     Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 10.1.1.5:100
```

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

```
RP/0/RSP0/CPU0: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                   Segment 1                       Segment 2
Group      Name       ST   Description            ST       Description            ST
------------------------   ------------------------------   ------------------------------
evpn_vpws_group
           evpn_vpws_100
                      UP   Gi0/0/0/3.2500         UP       EVPN 100,11,10.1.1.5    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 10.1.1.5 (node A).

```
RP/0/RSP0/CPU0: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 0x1200008; interworking none
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
      drops: illegal VLAN 0, illegal length 0
  EVPN: neighbor 10.1.1.5, PW ID: evi 100, ac-id 11, state is up ( established )
    XC ID 0xa0000003
    Encapsulation MPLS
    Source address 10.1.1.2
    Encap type Ethernet, control word enabled
    Sequencing not set
    Preferred path Active : SR TE srte_c_10000_ep_10.1.1.5, On-Demand, fallback enabled
    Tunnel : Up
    Load Balance Hashing: src-dst-mac

      EVPN         Local                          Remote
      ------------ ------------------------------ ------------------------------
      Label        80056                          80040
      MTU          1500                           1500
      Control word enabled                        enabled
      AC ID        21                             11
      EVPN type    Ethernet                       Ethernet

      ------------ ------------------------------ ------------------------------
    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 (10.1.1.5).

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

 Color             Endpoint  Admin   Oper              Binding
                             State  State                 SID
------ -------------------- ------ ------ --------------------
 10000              10.1.1.5    up     up                80058
```

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 (10.1.1.207).

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_10.1.1.5) is link-disjoint from LSP at Node D (srte_c_10000_ep_10.1.1.6).

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

SR-TE policy database
---------------------

Color: 10000, End-point: 10.1.1.5
  Name: srte_c_10000_ep_10.1.1.5
  Status:
    Admin: up  Operational: up for 00:12:35 (since Jul 10 19:49:21.890)
  Candidate-paths:
```

```
    Preference: 200 (BGP ODN) (shutdown)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_10000_ep_10.1.1.5_discr_200
        PLSP-ID: 7
      Dynamic (invalid)
    Preference: 100 (BGP ODN) (active)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_10000_ep_10.1.1.5_discr_100
        PLSP-ID: 6
      Dynamic (pce 10.1.1.207) (valid)
        Metric Type: IGP,   Path Accumulated Metric: 40
          16007 [Prefix-SID, 10.1.1.7]
          16008 [Prefix-SID, 10.1.1.8]
          80005 [Adjacency-SID, 11.5.8.8 - 11.5.8.5]
  Attributes:
    Binding SID: 80058
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

### 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 10.1.1.4:101). The output includes an EVPN route-type 1 route with color 10000 originated at Node B (10.1.1.6).

```
RP/0/RSP0/CPU0:Node-D# show bgp l2vpn evpn rd 10.1.1.4:101
BGP router identifier 10.1.1.4, 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 570
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
              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: 10.1.1.4:101 (default for vrf VPWS:101)
*>i[1][0000.0000.0000.0000.0000][12]/120
                      10.1.1.6 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/CPU0:Node-D# show bgp l2vpn evpn rd 10.1.1.4:101
[1][0000.0000.0000.0000.0000][12]/120
BGP routing table entry for [1][0000.0000.0000.0000.0000][12]/120, Route Distinguisher:
10.1.1.4:101
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
    10.1.1.6 C:10000 (bsid:80047) (metric 40) from 10.1.1.253 (10.1.1.6)
      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: 10.1.1.6, Cluster list: 10.1.1.253
      SR policy color 10000, up, registered, bsid 80047, if-handle 0x00001720

     Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 10.1.1.6:101
```

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

```
RP/0/RSP0/CPU0:Node-D# 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                   Segment 1                       Segment 2
Group      Name       ST   Description            ST       Description             ST
------------------------   ------------------------------   ------------------------------
evpn_vpws_group
           evpn_vpws_101
                      UP   Gi0/0/0/1.2500         UP       EVPN 101,12,10.1.1.6    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 10.1.1.6 (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: GigabitEthernet0/0/0/1.2500, 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 10.1.1.6, PW ID: evi 101, ac-id 12, state is up ( established )
    XC ID 0xa000000d
    Encapsulation MPLS
    Source address 10.1.1.4
    Encap type Ethernet, control word enabled
    Sequencing not set
    Preferred path Active : SR TE srte_c_10000_ep_10.1.1.6, On-Demand, fallback enabled
    Tunnel : Up
    Load Balance Hashing: src-dst-mac

      EVPN         Local                          Remote
      ------------ ------------------------------ ------------------------------
      Label        80045                          80060
      MTU          1500                           1500
```

```
    Control word enabled                        enabled
    AC ID        22                             12
    EVPN type    Ethernet                       Ethernet

    ------------ ------------------------------ ------------------------------
  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 (10.1.1.6).

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

 Color             Endpoint  Admin   Oper              Binding
                             State  State                 SID
------ -------------------- ------ ------ --------------------
 10000              10.1.1.6     up     up                80047
```

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 (10.1.1.207).

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_10.1.1.6) is link-disjoint from LSP at Node C (srte_c_10000_ep_10.1.1.5).

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

SR-TE policy database
---------------------

Color: 10000, End-point: 10.1.1.6
  Name: srte_c_10000_ep_10.1.1.6
  Status:
    Admin: up  Operational: up for 01:23:04 (since Jul 10 18:42:07.350)
  Candidate-paths:
    Preference: 200 (BGP ODN) (shutdown)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_10000_ep_10.1.1.6_discr_200
        PLSP-ID: 17
      Dynamic (invalid)
    Preference: 100 (BGP ODN) (active)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_10000_ep_10.1.1.6_discr_100
        PLSP-ID: 16
      Dynamic (pce 10.1.1.207) (valid)
        Metric Type: IGP,   Path Accumulated Metric: 40
          16001 [Prefix-SID, 10.1.1.1]
          16006 [Prefix-SID, 10.1.1.6]
  Attributes:
    Binding SID: 80047
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

# Manually Provisioned SR Policy

Manually provisioned SR policies are configured on the head-end router. These policies can use dynamic paths or explicit paths. See the SR-TE Policy Path Types, on page 203 section for information on manually provisioning an SR policy using dynamic or explicit paths.

# PCE-Initiated SR Policy

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 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).

For more information, see the PCE-Initiated SR Policies, on page 302 section.

# Cumulative Metric Bounds (Delay-Bound Use-Case)

***Table 19: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Cumulative Metric Bounds (Delay-Bound use-case) | Release 7.3.1 | With this feature, SRTE calculates a shortest path that satisfies multiple metric bounds.<br>This feature provides flexibility for finding paths within metric bounds, for parameters such as latency, hop count, IGP and TE. |

SRTE can calculate a shortest path with cumulative metric bounds. For example, consider these metric bounds:

- IGP metric <= 10
- TE metric <= 60
- Hop count <= 4
- Latency <= 55

When an SR policy is configured on a head-end node with these metric bounds, a path is finalized towards the specified destination only if it meets each of these criteria.

You can set the maximum number of attempts for computing a shortest path that satisfies the cumulative metric bounds criteria, by using the **kshortest-paths** command in SR-TE configuration mode.

### Restrictions

- PCE-based cumulative metric bounds computations are not supported. You must use non-PCE (SR-TE topology) based configuration for path calculation, for cumulative bounds.
- If you use PCE dynamic computation configuration with cumulative bounds, the PCE computes a path and validates against cumulative bounds. If it is valid, then the policy is created with this path on PCC. If the initial path doesn't respect the bounds, then the path is not considered, and no further K-shortest path algorithm is executed to find the path.

### Configuring SRTE Shortest Path Calculation For Cumulative Metric Bounds

You can enable this feature for SR, and ODN SR policy configurations, as shown below.

### SR Policy

**SR Policy** - A policy called **fromAtoB_XTC** is created towards destination IP address 192.168.0.2. Also, the candidate-paths preference, and other attributes are enabled.

```
Router# configure terminal
Router(config)# segment-routing traffic-eng policy fromAtoB_XTC
Router(config-sr-te-policy)# color 2 end-point ipv4 192.168.0.2
Router(config-sr-te-policy)# candidate-paths preference 100
Router(config-sr-te-policy-path-pref)# dynamic metric type te
```

**Cumulative Metric bounds** – IGP, TE, hop count, and latency metric bounds are set. SRTE calculates paths only when each criterion is satisfied.

```
Router(config-sr-te-policy-path-pref)# constraints bounds cumulative
Router(config-sr-te-pref-const-bounds-type)# type igp 10
Router(config-sr-te-pref-const-bounds-type)# type te 60
Router(config-sr-te-pref-const-bounds-type)# type hopcount 4
Router(config-sr-te-pref-const-bounds-type)# type latency 55
Router(config-sr-te-pref-const-bounds-type)# commit
```

### ODN SR Policy

**SR ODN Policy** – An SR ODN policy with color 1000 is created. Also, the candidate-paths value is on-demand.

```
Router# configure terminal
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color 1000 dynamic metric type te
Router(config-sr-te)# candidate-paths on-demand
Router(config-sr-te-candidate-path-type)# exit
Router(config-sr-te-candidate-path)# exit
```

**Cumulative Metric bounds** – IGP, TE, hop count, and latency metric bounds are set for the policy. SRTE calculates paths, only when each criterion is satisfied.

```
Router(config-sr-te)# on-demand color 1000 dynamic bounds cumulative
Router(config-sr-te-odc-bounds-type)# type igp 100
Router(config-sr-te-odc-bounds-type)# type te 60
Router(config-sr-te-odc-bounds-type)# type hopcount 6
Router(config-sr-te-odc-bounds-type)# type latency 1000
Router(config-sr-te-odc-bounds-type)# commit
```

To set the maximum number of attempts for computing paths that satisfy the cumulative metric bounds criteria, use the **kshortest-paths** command.

```
Router# configure terminal
Router(config)# segment-routing traffic-eng
```

```
Router(config-sr-te)# kshortest-paths 120
Router(config-sr-te)# commit
```

### Verification

Use this command to view SR policy configuration details. Pointers:

- The **Number of K-shortest-paths** field displays 4. It means that the K-shortest path algorithm took 4 computations to find the right path. The 4 shortest paths that are computed using K-shortest path algorithm did not respect the cumulative bounds. The fifth shortest path is valid against the bounds.
- The values for the metrics of the actual path (**TE**, **IGP**, **Cumulative Latency** and **Hop count** values in the **Dynamic** section) are within the configured cumulative metric bounds.

```
Router# show segment-routing traffic-eng policy color 2

Color: 2, End-point: 192.168.0.2
  Name: srte_c_2_ep_192.168.0.2
  Status:
    Admin: up  Operational: up for 3d02h (since Dec 15 12:13:21.993)

  Candidate-paths:

    Preference: 100 (configuration) (active)

      Name: fromAtoB_XTC
      Requested BSID: dynamic
      Constraints:
        Protection Type: protected-preferred
        Affinity:
          exclude-any:
          red
        Maximum SID Depth: 10
        IGP Metric Bound: 10
        TE Metric Bound: 60
        Latency Metric Bound: 55
        Hopcount Metric Bound: 4

      Dynamic (valid)

        Metric Type: TE,   Path Accumulated Metric: 52
        Number of K-shortest-paths: 4
        TE Cumulative Metric: 52
        IGP Cumulative Metric: 3
        Cumulative Latency: 52
        Hop count: 3
          16004 [Prefix-SID, 192.168.0.4]
          24003 [Adjacency-SID, 16.16.16.2 - 16.16.16.5]
          24001 [Adjacency-SID, 14.14.14.5 - 14.14.14.4]

  Attributes:

    Binding SID: 24011
    Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
```

# SR-TE BGP Soft Next-Hop Validation For ODN Policies

*Table 20: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR-TE BGP Soft Next-Hop Validation For ODN Policies | Release 7.3.2 | This feature addresses BGP Next-Hop reachability issues through BGP Next-Hop *soft* validation, and also enhances BGP best path selection.<br>New commands:<br>• **nexthop validation color-extcomm disable**<br>• **nexthop validation color-extcomm sr-policy**<br>• **bgp bestpath igp-metric sr-policy** |

Before a BGP router installs a route in the routing table, it checks its own reachability to the Next-Hop (NH) IP address of the route. In an SR-TE domain, a NH address may not be redistributed within the AS, or to a neighbor AS. So, BGP cannot reach the NH, and does not install the corresponding route into the routing table. The following workarounds are available, but they are tedious and might impact scalability:

1. Enable a non-default, static route to null0 covering the routes
2. Inject the routes into BGP using BGP-Labeled Unicast configuration
3. Redistribute routes between IGP domains

This feature introduces a more optimal design and solution - When you enable an SR policy on the SR-TE headend router, configure the nexthop validation color-extcomm sr-policy command in BGP configuration mode. It instructs BGP that, instead of NH reachability validation of BGP routes, the validation is done for SR policy-installed color NH addresses. When the NH address of such a route is reachable, the route is added to the routing table.

Also, this configuration on the ingress/headend PE router reduces the route scale for NH reachability, and service (VPN) routes automatically get NH reachability.

RR configuration – For intermediate router configuration, enable the RR with the nexthop validation color-extcomm disable command. When enabled, and L3VPN prefixes are associated with a color ID, BGP skips NH validation on the RR.

When the RR has no reachability to the color-extcomm NH, either enable this command, or use a legacy static route.

The following sequence occurs when the headend router receives L3VPN prefixes based on a color ID such as purple, green, etc.

1. The router checks/learns the local SR policy, or requests the ODN SR policy for color ID and NH

2. BGP does validation of the SR policy routes' NH addresses and applies the corresponding NH AD/metric. For a NH with a specific BGP-based color attribute, SR-PCE provides the AD/metric

   With BGP NH reachability, traffic is transported smoothly

3. On the RR, BGP does not validate NH reachability

### BGP Best Path Selection Based On SR Policy Effective Metric

BGP uses an algorithm to select the best path for installing the route in the RIB or for making a choice of which BGP path to propagate. At a certain point in the process, if there is IGP reachability to a BGP NH address, the algorithm chooses the path with the lowest IGP metric as the best path. The SR Policy path metric is not considered even if it has a better metric. This feature addresses the issue.

To ensure that BGP prefers the SR policy path metric over the IGP metric, enable bgp bestpath igp-metric sr-policy in BGP configuration mode.

### Configurations

**Configuring BGP Soft Next-Hop Validation** (Headend Router)

```
Headend # configure
Headend (config) # router bgp 100
Headend (config-bgp)# nexthop validation color-extcomm sr-policy
Headend (config-bgp)# commit
Headend (config-bgp)# end
```

**Configuring BGP Soft Next-Hop Validation** (Route Reflector)

```
RR # configure
RR (config) # router bgp 100
RR (config-bgp)# nexthop validation color-extcomm disable
RR (config-bgp)# commit
RR (config-bgp)# end
```

**Configuring BGP Best Path Selection Based on SR Policy Metric** (Headend Router)

```
Headend # configure
Headend (config) # router bgp 100
Headend (config-bgp)# bgp bestpath igp-metric sr-policy
Headend (config-bgp)# commit
Headend (config-bgp)# end
```

### Verification

Use this command to view BGP Soft Next-Hop Validation details.

```
Headend # show bgp process detail | i Nexthop
Use SR-Policy admin/metric of color-extcomm Nexthop during path comparison: enabled ExtComm
 Color Nexthop validation: SR-Policy then RIB
```

Use this command to view BGP Best Path Selection Based on SR Policy Metric.

```
Headend # show bgp vrf VRF1002 ipv4 unicast 207.77.2.0

BGP routing table entry for 207.77.2.0/24, Route Distinguisher: 18522:1002 Versions:
Process bRIB/RIB SendTblVer
Speaker 5232243 5232243 Paths: (1 available, best #1)
Advertised to CE peers (in unique update groups): 10.11.2.11 101.15.2.2
Path #1: Received by speaker 0

Advertised to CE peers (in unique update groups): 10.11.2.11 101.15.2.2
```

```
16611 770
10.1.1.33 C:1129 (bsid:27163) (admin 20) (metric 25) from 10.1.1.100 (10.1.1.33)
Received Label 24007
Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported
Received Path ID 1, Local Path ID 1, version 5232243
Extended community: Color:1129 RT:17933:1002 RT:18522:1002
Originator: 10.1.1.33, Cluster list: 10.1.1.100
SR policy color 1129, up, registered, bsid 27163, if-handle 0x200053dc
Source AFI: VPNv4 Unicast, Source VRF: default, Source Route Distinguisher: 18522:3002
```

**Details**

- **10.1.1.33 C:1129** - BGP path is selected based on the SR policy with color ID C:1129
- If no SR policy is up, or if the SR policy metric is not configured, only the RIB metric is displayed
- **admin 20** and **metric 25** are SR policy references

# SR-TE Policy Path Types

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* 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 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.

A candidate path has the following characteristics:

- It has a preference – If two policies have the 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.

# Dynamic Paths

### Behaviors and Limitations

For a dynamic path that traverses a specific interface between nodes (segment), the algorithm may encode this segment using an Adj-SID. The SR-TE process prefers the protected Adj-SID of the link, if one is available. In addition, the SR-TE process prefers a manual protected Adj-SID over a dynamic protected Adj-SID.

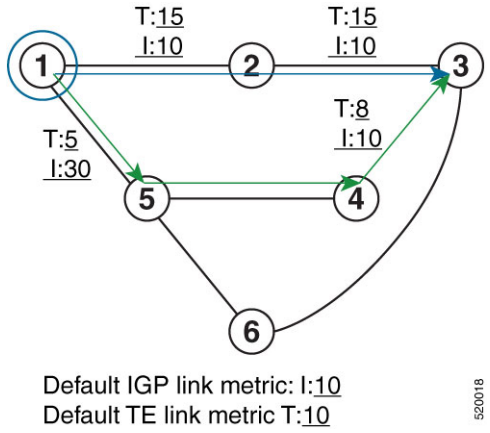
You can configure the path to prefer the protected or unprotected Adj-SID, or to use only protected or unprotected Adj-SID. See Segment Protection-Type Constraint, on page 207.

## Optimization Objectives

Optimization objectives allow the head-end router to compute a SID-list that expresses the shortest dynamic path according to the selected metric type:

- IGP metric — Refer to the "Implementing IS-IS" and "Implementing OSPF" chapters in the *Routing Configuration Guide for Series Routers*.
- TE metric — See the Configure Interface TE Metrics, on page 204 section for information about configuring TE metrics.

This example shows a dynamic path from head-end router 1 to end-point router 3 that minimizes IGP or TE metric:




- The blue path uses the minimum IGP metric: Min-Metric (1 → 3, IGP) = SID-list <16003>; cumulative IGP metric: 20
- The green path uses the minimum TE metric: Min-Metric (1 → 3, TE) = SID-list <16005, 16004, 16003>; cumulative TE metric: 23

### 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
Router(config-sr-te)# interface type interface-path-id
Router(config-sr-te-if)# metric value
```

#### 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
```

## Constraints

Constraints allow the head-end router to compute a dynamic path according to the selected metric type:

- Affinity — You can apply a color or name to links or interfaces by assigning affinity bit-maps to them. You can then specify an affinity (or relationship) between an SR policy path and link colors. SR-TE computes a path that includes or excludes links that have specific colors,or combinations of colors. See the Named Interface Link Admin Groups and SR-TE Affinity Maps, on page 205 section for information on named interface link admin groups and SR-TE Affinity Maps.
- Disjoint — SR-TE computes a path that is disjoint from another path in the same disjoint-group. Disjoint paths do not share network resources. Path disjointness may be required for paths between the same pair of nodes, between different pairs of nodes, or a combination (only same head-end or only same end-point).
- Flexible Algorithm — Flexible Algorithm allows for user-defined algorithms where the IGP computes paths based on a user-defined combination of metric type and constraint.
- Protection type — For a dynamic path that traverses a specific interface between nodes (segment), or for an explicit path using IP addresses of intermediate links, the algorithm may encode this segment using an Adj-SID. You can specify the path to prefer protected or unprotected Adj-SIDs, or to use only protected or unprotected Adj-SIDs. See Segment Protection-Type Constraint, on page 207 for information about configuring the protection type.

### Named Interface Link Admin Groups and SR-TE Affinity Maps

Named Interface Link Admin Groups and SR-TE Affinity Maps provide 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).

Named Interface Link Admin Groups and SR-TE Affinity Maps let you assign, or map, up to 256 color names for affinity and attribute-flag attributes instead of 32-bit hexadecimal numbers. After mappings are defined, the attributes can be referred to by the corresponding color name in the CLI. Furthermore, you can define constraints using *include-any*, *include-all*, and *exclude-any* arguments, where each statement can contain up to 10 colors.

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

*Configure Named 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 TenGigE0/0/1/2
Router(config-sr-if)# affinity
Router(config-sr-if-affinity)# name RED
```

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 RED bit-position 23
```

### Configuring Link Admin Group: Example

The following example shows how to assign affinity to interfaces and to define affinity maps. This configuration is applicable to any router (SR-TE head-end or transit node) with colored interfaces.

```
segment-routing
 traffic-eng
  interface TenGigE0/0/1/1
   affinity
    name CROSS
    name RED
   !
  !
  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
```

*Segment Protection-Type Constraint*

***Table 21: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Segment Protection-Type Constraint | Release 7.4.1 | This feature introduces the ability to control whether protected or unprotected segments are used when encoding the SID-list of an SR policy candidate path.<br><br>The types of segments that could be used when building a SID-list include prefix SIDs and adjacency SIDs. |

This feature introduces the ability to control whether protected or unprotected segments are used when encoding the SID-list of an SR policy candidate path. The types of segments that could be used when building a SID-list include prefix SIDs and adjacency SIDs.

A prefix SID is a global segment representing a prefix that identifies a specific node. A prefix SID is programmed with a backup path computed by the IGP using TI-LFA.

An adjacency SID is a local segment representing an IGP adjacency. An adjacency SID can be programmed with or without protection. Protected adjacency SIDs are programmed with a link-protectant backup path computed by the IGP (TI-LFA) and are used if the link associated with the IGP adjacency fails.

Prefix SIDs and adjacency SIDs can be leveraged as segments in a SID-list in order to forward a packet along a path traversing specific nodes and/or over specific interfaces between nodes. The type of segment used when encoding the SID-list will determine whether failures along the path would be protected by TI-LFA. Depending on the offering, an operator may want to offer either unprotected or protected services over traffic engineered paths.

The following behaviors are available with the segment protection-type constraint:

- **protected-only** — The SID-list must be encoded using protected segments.
- **protected-preferred** — The SID-list should be encoded using protected segments if available; otherwise, the SID-list may be encoded using unprotected Adj-SIDs. This is the default behavior when no segment protection-type constraint is specified.
- **unprotected-only** — The SID-list must be encoded using unprotected Adj-SID.
- **unprotected-preferred** — The SID-list should be encoded using unprotected Adj-SID if available, otherwise SID-list may be encoded using protected segments.

### Usage Guidelines and Limitations

Observe the following guidelines and limitations for the platform:

- This constraint applies to candidate-paths of manual SR policies with either dynamically computed paths or explicit paths.
- This constraint applies to On-Demand SR policy candidate-paths.

- PCEP has been augmented (vendor-specific object) to allow a PCC to indicate the segment protection-type constraint to the PCE.
- When the segment protection type constraint is protected-only or unprotected-only, the path computation must adhere to the constraint. If the constraint is not satisfied, the SR policy will not come up on such candidate path.
- When the segment protection-type constraint is unprotected-only, the entire SID-list must be encoded with unprotected Adj-SIDs.
- When the segment protection-type constraint is protected-only, the entire SID-list must be encoded with protected Adj-SIDs or Prefix SIDs.

### Configuring Segment Protection-Type Constraint

Use the **constraints segments protection** {**protected-only** | **protected-preferred** | **unprotected-only** | **unprotected-preferred**} command to configure the segment protection-type behavior.

The following example shows how to configure the policy with a SID-list that must be encoded using protected segments:

```
Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# color 10 end-point ipv4 10.1.1.4
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# constraints
Router(config-sr-te-path-pref-const)# segments
Router(config-sr-te-path-pref-const-seg)# protection protected-only
```

## Configure SR Policy with Dynamic Path

To configure a SR-TE policy with a dynamic path, optimization objectives, and affinity constraints, complete the following configurations:

1. Define the optimization objectives. See the Optimization Objectives, on page 204 section.
2. Define the constraints. See the Constraints, on page 205 section.
3. Create the policy.

### Behaviors and Limitations

For a dynamic path that traverses a specific interface between nodes (segment), the algorithm may encode this segment using an Adj-SID. The SR-TE process prefers the protected Adj-SID of the link, if one is available. In addition, the SR-TE process prefers a manual protected Adj-SID over a dynamic protected Adj-SID.

You can configure the path to prefer protected or unprotected segments, or to use only protected or unprotected segments.

### Examples

The following example shows a configuration of an SR policy at an SR-TE head-end router. The policy has a dynamic path with optimization objectives and affinity constraints computed by the head-end router.

```
segment-routing
 traffic-eng
  policy foo
   color 100 end-point ipv4 10.1.1.2
```

```
   candidate-paths
    preference 100
     dynamic
      metric
       type te
      !
     !
     constraints
      affinity
       exclude-any
        name RED
       !
      !
     !
    !
   !
  !
```

The following example shows a configuration of an SR policy at an SR-TE head-end router. The policy has a dynamic path with optimization objectives and affinity constraints computed by the SR-PCE.

```
segment-routing
 traffic-eng
  policy baa
   color 101 end-point ipv4 10.1.1.2
   candidate-paths
    preference 100
     dynamic
      pcep
      !
      metric
       type te
      !
     !
     constraints
      affinity
       exclude-any
        name BLUE
       !
      !
     !
    !
   !
  !
```

The following example shows a configuration of an SR policy at an SR-TE head-end router. The policy has a dynamic path with optimization objective and segment protection-type constraint computed by the head-end router.

```
segment-routing
 traffic-eng
  policy baa
   color 101 end-point ipv4 10.1.1.2
   candidate-paths
    preference 100
     dynamic
      metric
       type te
      !
     !
     constraints
      segments
       protection protected-only
       !
```

```
      !
     !
    !
   !
  !
```

The following example shows a configuration of an SR policy at an SR-TE head-end router. The policy has a dynamic path with optimization objective and segment protection-type constraint computed by the SR-PCE.

```
segment-routing
 traffic-eng
  policy baa
   color 101 end-point ipv4 10.1.1.2
   candidate-paths
    preference 100
     dynamic
      pcep
      !
      metric
       type te
      !
     !
     constraints
      segments
       protection protected-only
       !
      !
     !
    !
   !
  !
```

# Explicit Paths

## SR-TE Policy with Explicit Path

An explicit segment list is defined as a sequence of one or more segments. A segment can be configured as an IP address or an MPLS label representing a node or a link.

An explicit segment list can be configured with the following:

- IP-defined segments
- MPLS label-defined segments
- A combination of IP-defined segments and MPLS label-defined segments

### Usage Guidelines and Limitations

- An IP-defined segment can be associated with an IPv4 address (for example, a link or a Loopback address).
- When a segment of the segment list is defined as an MPLS label, subsequent segments can only be configured as MPLS labels.
- When configuring an explicit path using IP addresses of links along the path, the SR-TE process prefers the protected Adj-SID of the link, if one is available. In addition, when manual Adj-SIDs are configured, the SR-TE process prefers a manual protected Adj-SID over a dynamic protected Adj-SID.

- You can configure the path to prefer the protected or unprotected Adj-SID, or to use only protected or unprotected Adj-SID. See Segment Protection-Type Constraint, on page 207.

### Configure Local SR-TE Policy Using Explicit Paths

To configure an SR-TE policy with an explicit path, complete the following configurations:

1. Create the segment list.
2. Create the SR-TE policy.

Create a segment list with IPv4 addresses:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list name SIDLIST1
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls adjacency 10.1.1.3
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
```

Create a segment list with MPLS labels:

```
Router(config-sr-te)# segment-list name SIDLIST2
Router(config-sr-te-sl)# index 10 mpls label 16002
Router(config-sr-te-sl)# index 20 mpls label 16003
Router(config-sr-te-sl)# index 30 mpls label 16004
Router(config-sr-te-sl)# exit
```

Create a segment list with IPv4 addresses and MPLS labels:

```
Router(config-sr-te)# segment-list name SIDLIST3
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls label 16003
Router(config-sr-te-sl)# index 30 mpls label 16004
Router(config-sr-te-sl)# exit
```

Create the SR-TE policy:

```
Router(config-sr-te)# policy POLICY2
Router(config-sr-te-policy)# color 20 end-point ipv4 10.1.1.4
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 200
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST2
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST1
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
```

### Running Configuration

```
Router# show running-configuration

segment-routing
 traffic-eng
  segment-list SIDLIST1
```

```
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls adjacency 10.1.1.3
   index 30 mpls adjacency 10.1.1.4
  !
  segment-list SIDLIST2
   index 10 mpls label 16002
   index 20 mpls label 16003
   index 30 mpls label 16004
  !
  segment-list SIDLIST3
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls label 16003
   index 30 mpls label 16004
  !
  policy POLICY2
   color 20 end-point ipv4 10.1.1.4
   candidate-paths
    preference 200
     explicit segment-list SIDLIST2
     !
    !
    preference 100
     explicit segment-list SIDLIST1
     !
    !
   !
  !!
!
```

### Verification

Verify the SR-TE policy configuration using:

```
Router# show segment-routing traffic-eng policy name srte_c_20_ep_10.1.1.4

SR-TE policy database
---------------------

Color: 20, End-point: 10.1.1.4
  Name: srte_c_20_ep_10.1.1.4
  Status:
    Admin: up  Operational: up for 00:00:15 (since Jul 14 00:53:10.615)
  Candidate-paths:
    Preference: 200 (configuration) (active)
      Name: POLICY2
      Requested BSID: dynamic
        Protection Type: protected-preferred
        Maximum SID Depth: 8
      Explicit: segment-list SIDLIST2 (active)
        Weight: 1, Metric Type: TE
          16002
          16003
          16004

    Preference: 100 (configuration) (inactive)
      Name: POLICY2
      Requested BSID: dynamic
        Protection Type: protected-preferred
        Maximum SID Depth: 8
      Explicit: segment-list SIDLIST1 (inactive)
        Weight: 1, Metric Type: TE
          [Adjacency-SID, 10.1.1.2 - <None>]
          [Adjacency-SID, 10.1.1.3 - <None>]
```

```
      [Adjacency-SID, 10.1.1.4 - <None>]
 Attributes:
 Binding SID: 51301
Forward Class: Not Configured
 Steering labeled-services disabled: no
 Steering BGP disabled: no
 IPv6 caps enable: yes
 Invalidation drop enabled: no
```

## Configuring Explicit Path with Affinity Constraint Validation

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

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

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


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


/* Associate affinity constraints for SR-TE policies
Router(config-sr-te)# segment-list name SIDLIST1
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls adjacency 2.2.2.23
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list name SIDLIST2
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list name SIDLIST3
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.5
```

```
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit


Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# color 20 end-point ipv4 10.1.1.4
Router(config-sr-te-policy)# binding-sid mpls 1000
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 200
Router(config-sr-te-policy-path-pref)# constraints affinity exclude-any red
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST1
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST2
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST3
```

### Running Configuration

```
Router# show running-configuration
segment-routing
 traffic-eng

  interface GigabitEthernet0/0/0/0
   affinity
    blue
   !
  !
  interface GigabitEthernet0/0/0/1
   affinity
    blue
    green
   !
  !


  segment-list name SIDLIST1
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls adjacency 2.2.2.23
   index 30 mpls adjacency 10.1.1.4
  !
  segment-list name SIDLIST2
   index 10 mpls adjacency 10.1.1.2
   index 30 mpls adjacency 10.1.1.4
  !
  segment-list name SIDLIST3
   index 10 mpls adjacency 10.1.1.5
   index 30 mpls adjacency 10.1.1.4
  !
  policy POLICY1
   binding-sid mpls 1000
   color 20 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     explicit segment-list SIDLIST3
     !
    !
    preference 200
     explicit segment-list SIDLIST1
     !
     explicit segment-list SIDLIST2
     !
```

```
    constraints
     affinity
      exclude-any
       red
      !
     !
    !
   !
  !
  affinity-map
   blue bit-position 0
   green bit-position 1
   red bit-position 2
  !
 !
!
```

# Configure Explicit Path with Segment Protection-Type Constraint

For an SR policy with an explicit path that includes IP addresses of links, the SR-TE process encodes these segments using the corresponding adjacency SID (Adj-SID) for each link. The type of Adj-SID used (protected or unprotected) is determined by the segment protection-type constraint configured under the SR policy. See the Segment Protection-Type Constraint, on page 207.

## Configure Local SR-TE Policy Using Explicit Paths

Create a segment list with IP addresses:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list name SIDLIST1
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls adjacency 10.1.1.3
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
```

Create the SR-TE policy with segment protection-type constraint:

```
Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# color 20 end-point ipv4 10.1.1.4
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 SIDLIST1
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# constraints
Router(config-sr-te-path-pref-const)# segments
Router(config-sr-te-path-pref-const-seg)# protection protected-only
Router(config-sr-te-path-pref-const-seg)# commit
```

## Running Configuration

```
Router# show running-configuration
segment-routing
 traffic-eng
  segment-list SIDLIST1
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls adjacency 10.1.1.3
   index 30 mpls adjacency 10.1.1.4
```

```
!

policy POLICY1
 color 10 end-point ipv4 10.1.1.4
 candidate-paths
  preference 100
   explicit segment-list SIDLIST1
   !
  constraints
   segments
    protection protected-only
   !
  !
!
```

# Protocols

## Path Computation Element Protocol

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.

### 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[keychain WORD]
```

#### Configure PCEP Authentication

TCP Message Digest 5 (MD5) authentication has been used for authenticating PCEP (TCP) sessions by using a clear text or encrypted password. This feature introduces support for TCP Authentication Option (TCP-AO), which replaces the TCP MD5 option.

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

- Protection against replays for long-lived TCP connections
- More details on the security association with TCP connections than TCP MD5
- A larger set of MACs with minimal system and operational changes

TCP-AO is compatible with Master Key Tuple (MKT) configuration. TCP-AO also protects connections when using the same MKT across repeated instances of a connection. TCP-AO protects the connections by using traffic key that are derived from the MKT, and then coordinates changes between the endpoints.

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

**TCP Message Digest 5 (MD5) Authentication**

Use the **password** {**clear** | **encrypted**} *LINE* command to enable TCP MD5 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

```
Router(config-sr-te-pcc)# pce address ipv4 PCE-address[password {clear | encrypted} LINE]
```

**TCP Authentication Option (TCP-AO)**

Use the **tcp-ao** *key-chain* [**include-tcp-options**] command to enable TCP Authentication Option (TCP-AO) authentication for all PCEP peers. Any TCP segment coming from the PCC that does not contain a MAC matching the configured key chain will be rejected. Use the **include-tcp-options** keyword to include other TCP options in the header for MAC calculation.

```
Router(config-sr-te-pcc)# pce address ipv4 PCE-address tcp-ao key-chain [include-tcp-options]
```

### 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 at 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 (12).

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

**Note** The platform's SR-TE label imposition capabilities are as follows:

- Up to 12 transport labels when no service labels are imposed
- Up to 9 transport labels when service labels are imposed

For cases with path computation at PCE, 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.

**Note** If the configured MSD values are different, the per-LSP MSD takes precedence over the per-node MSD.

After path computation, the resulting label stack size is verified against the MSD requirement.

- If the label stack size is larger than the MSD and path computation is performed by PCE, then the PCE returns a "no path" response to the PCC.
- If the label stack size is larger than the MSD and path computation is performed by PCC, then the PCC will not install the path.

**Note** A sub-optimal path (if one exists) that satisfies the MSD constraint could be computed in the following cases:

- For a dynamic path with TE metric, when the PCE is configured with the **pce segment-routing te-latency** command or the PCC is configured with the **segment-routing traffic-eng te-latency** command.
- For a dynamic path with LATENCY metric
- For a dynamic path with affinity constraints

For example, if the PCC MSD is 4 and the optimal path (with an accumulated metric of 100) requires 5 labels, but a sub-optimal path exists (with accumulated metric of 110) requiring 4 labels, then the sub-optimal path is installed.

### Customize the SR-TE Path Calculation

Use the **te-latency** command to enable ECMP-aware path computation for TE metric.

```
Router(config-sr-te)# te-latency
```

**Note** ECMP-aware path computation is enabled by default for IGP and LATENCY metrics.

### Configure PCEP Redundancy Type

Use the **redundancy pcc-centric** command to enable PCC-centric high-availability model. The PCC-centric model changes the default PCC delegation behavior to the following:

- After LSP creation, LSP is automatically delegated to the PCE that computed it.
- If this PCE is disconnected, then the LSP is redelegated to another PCE.
- If the original PCE is reconnected, then the delegation fallback timer is started. When the timer expires, the LSP is redelegated back to the original PCE, even if it has worse preference than the current PCE.

```
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 10.1.1.57 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.

```
segment-routing
 traffic-eng
  pcc
   source-address ipv4 10.1.1.2
   pce address ipv4 10.1.1.57
    precedence 150
    password clear <password>
   !
   pce address ipv4 10.1.1.58
    precedence 200
    password clear <password>
   !
   pce address ipv4 10.1.1.59
    precedence 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: 10.1.1.57, Precedence: 150, (best PCE)
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation
```

```
Peer address: 10.1.1.58, Precedence: 200
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 10.1.1.59, Precedence: 250
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation
```

## Configure SR-TE PCE Groups

*Table 22: Feature History Table*

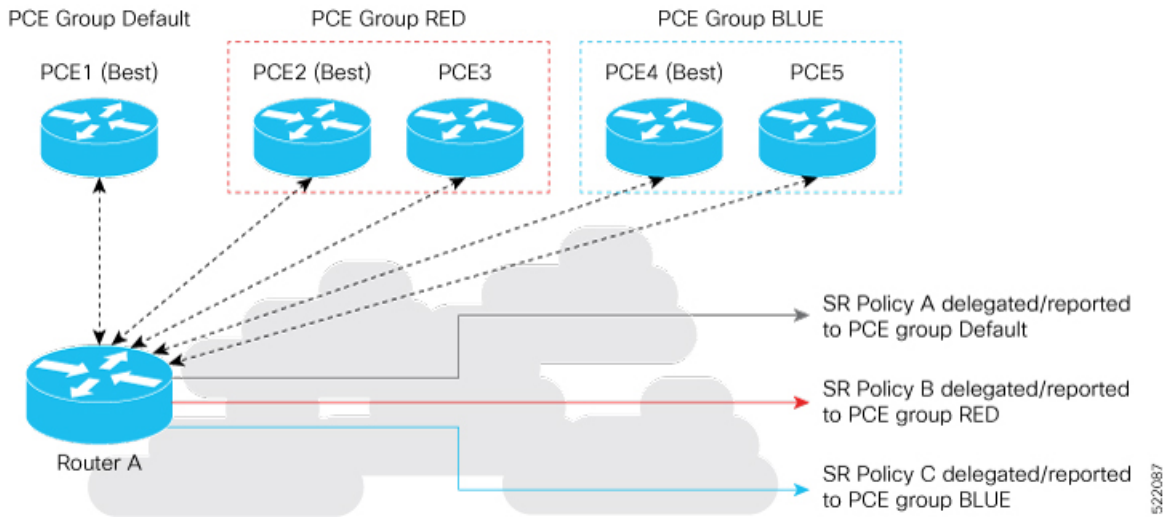
| Feature Name | Release | Description |
|---|---|---|
| SR-TE PCE Groups | Release 7.3.2 | This feature allows an SR policy to be delegated to a set of PCE servers configured under a PCE group. Multiple PCE groups can be configured to allow SR policies on the same head-end to be delegated to different sets of PCEs.<br>With this functionality, an operator can designate sets of PCEs for various purposes, such as PCE-per-service-type or PCE-per-wholesale-customers. |

This feature allows an SR policy to be delegated or reported to a set of PCE servers configured under a PCE group. Multiple PCE groups can be configured to allow different SR policies on the same head-end to be delegated or reported to different sets of PCEs.

With this functionality, an operator can designate sets of PCEs for various purposes, such as PCE-per-service-type or PCE-per-wholesale-customer.

In the figure below, Router A has a PCEP session with 5 PCEs. The PCEs are configured over 3 PCE groups. PCE1 is in the “default” group. PCE2 and PCE3 are in the RED group. PCE4 and PCE5 are in the BLUE group.

*Figure 12: Example: PCE Groups*




In case of PCE failure, each candidate path is re-delegated to the next-best PCE within the same PCE group. For example, if the best PCE in the RED group (PCE2) fails, then all candidate paths in the RED group fallback to the secondary PCE in the RED group (PCE3). If all the PCEs in the RED group fail, then all candidate paths in the RED group become undelegated; they are not delegated to the PCEs in the BLUE group. If there are no more available PCEs in the given PCE group, then the outcome is the same as when there are no available PCEs.

### Configure PCE Groups

Use the **segment-routing traffic-eng pcc pce address** {**ipv4** *ipv4_addr* | **ipv6** *ipv6_addr*} **pce-group** *WORD* command to configure the PCE groups.

The following example shows how to configure the PCE groups

- PCE1 in the “default” group
- PCE2 and PCE3 in the “red” group
- PCE4 and PCE5 in the “blue” group

```
Router(config)# segment-routing traffic-eng pcc
Router(config-sr-te-pcc)# pce address ipv4 10.1.1.1
Router(config-pcc-pce)# precedence 10
Router(config-pcc-pce)# exit

Router(config-sr-te-pcc)# pce address ipv4 2.2.2.2
Router(config-pcc-pce)# precedence 20
Router(config-pcc-pce)# pce-group red
Router(config-pcc-pce)# exit

Router(config-sr-te-pcc)# pce address ipv4 3.3.3.3
Router(config-pcc-pce)# precedence 30
Router(config-pcc-pce)# pce-group red
Router(config-pcc-pce)# exit

Router(config-sr-te-pcc)# pce address ipv4 4.4.4.4
Router(config-pcc-pce)# precedence 40
Router(config-pcc-pce)# pce-group blue
```

```
Router(config-pcc-pce)# exit

Router(config-sr-te-pcc)# pce address ipv4 5.5.5.5
Router(config-pcc-pce)# precedence 50
Router(config-pcc-pce)# pce-group blue
Router(config-pcc-pce)# exit
```

### Verification

```
segment-routing
 traffic-eng
  pcc
   pce address ipv4 10.1.1.1
    precedence 10
   !
   pce address ipv4 2.2.2.2
    precedence 20
    pce-group red
   !
   pce address ipv4 3.3.3.3
    precedence 30
    pce-group red
   !
   pce address ipv4 4.4.4.4
    precedence 40
    pce-group blue
   !
   pce address ipv4 5.5.5.5
    precedence 50
    pce-group blue
   !
  !
 !
!
```

### Assign PCE Group to a Candidate Path or ODN Template

Use the **segment-routing traffic-eng policy** *policy* **pce-group** *WORD* command to assign the PCE group to all candidate paths of an SR policy.

Use the **segment-routing traffic-eng policy** *policy* **candidate-paths preference** *pref* **pce-group** *WORD* command to assign the PCE group to a specific candidate path of an SR policy.

Use the **segment-routing traffic-eng on-demand color** *color* **pce-group** *WORD* command to assign the PCE group to on-demand candidate paths triggered by an ODN template.

**Note** Only one PCE group can be attached to a given SR policy candidate path.

The following example shows how to configure a policy with all candidate paths delegated/reported to PCEs in the default group:

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy A
Router(config-sr-te-policy)# color 100 end-point ipv4 192.168.0.2
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# dynamic
Router(config-sr-te-pp-info)# pcep
```

```
Router(config-sr-te-path-pcep)# exit
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# exit
Router(config-sr-te-policy)# exit
```

The following example shows how to configure a policy with all candidate paths delegated/reported to PCEs in the red group:

```
Router(config-sr-te)# policy B
Router(config-sr-te-policy)# color 100 end-point ipv4 192.168.0.3
Router(config-sr-te-policy)# pce-group red
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# dynamic
Router(config-sr-te-pp-info)# pcep
Router(config-sr-te-path-pcep)# exit
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# exit
Router(config-sr-te-policy)# exit
```

The following example shows how to configure a policy with a specific candidate path (explicit path) reported to PCEs in the blue group:

```
Router(config-sr-te)# policy C
Router(config-sr-te-policy)# color 100 end-point ipv4 192.168.0.4
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# pce-group blue
Router(config-sr-te-policy-path-pref)# explicit segment-list SLA
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# exit
Router(config-sr-te-policy)# exit
```

The following example shows how to configure an ODN template with on-demand candidate paths delegated/reported to PCEs in the blue group:

```
Router(config-sr-te)# on-demand color 10
Router(config-sr-te-color)# pce-group blue
Router(config-sr-te-color)# dynamic
Router(config-sr-te-color-dyn)#pcep
Router(config-sr-te-color-dyn-pce)#
```

### Running Config

```
segment-routing
 traffic-eng
  on-demand color 10
   dynamic
    pcep
    !
   !
   pce-group blue
  !
  policy A
   color 100 end-point ipv4 192.168.0.2
   candidate-paths
    preference 100
```

```
     dynamic
      pcep
      !
     !
    !
   !
  !
  policy B
   color 100 end-point ipv4 192.168.0.3
   pce-group red
   candidate-paths
    preference 100
     dynamic
      pcep
      !
     !
    !
   !
  !
  policy C
   color 100 end-point ipv4 192.168.0.4
   candidate-paths
    preference 100
     explicit segment-list SLA
     !
     pce-group blue
    !
   !
  !
 !
!
end
```

## Verification

```
Router# show segment-routing traffic-eng pcc ipv4 peer

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

Peer address: 10.1.1.1
  Precedence: 10 (best PCE)
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 2.2.2.2
  Group: red, Precedence 20
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 3.3.3.3
  Group: red, Precedence 30
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 4.4.4.4
  Group: blue, Precedence 40
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation

Peer address: 5.5.5.5
  Group: blue, Precedence 50
  State up
  Capabilities: Stateful, Update, Segment-Routing, Instantiation
```

```
Router# show segment-routing traffic-eng policy name srte_c_100_ep_192.168.0.3

SR-TE policy database
---------------------

Color: 100, End-point: 192.168.0.3
  Name: srte_c_100_ep_192.168.0.3
  Status:
    Admin: up  Operational: up for 00:13:26 (since Sep 17 22:52:48.365)
  Candidate-paths:
    Preference: 100 (configuration)
      Name: B
      Requested BSID: dynamic
      PCC info:
        Symbolic name: cfg_B_discr_100
        PLSP-ID: 2
        Protection Type: protected-preferred
        Maximum SID Depth: 10
       PCE Group: red
       Dynamic (pce 192.168.1.4) (valid)
        Metric Type: TE,   Path Accumulated Metric: 10
  Attributes:
    Forward Class: 0
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: no
    Invalidation drop enabled: no
```

# BGP SR-TE

BGP may be used to distribute SR Policy candidate paths to an SR-TE head-end. Dedicated BGP SAFI and NLRI have been defined to advertise a candidate path of an SR Policy. The advertisement of Segment Routing policies in BGP is documented in the IETF draft https://datatracker.ietf.org/doc/draft-ietf-idr-segment-routing-te-policy/

SR policies with IPv4 and IPv6 end-points can be advertised over BGPv4 or BGPv6 sessions between the SR-TE controller and the SR-TE headend.

The Cisco IOS-XR implementation supports the following combinations:

- IPv4 SR policy advertised over BGPv4 session
- IPv6 SR policy advertised over BGPv4 session
- IPv6 SR policy advertised over BGPv6 session

## Configure BGP SR Policy Address Family at SR-TE Head-End

Perform this task to configure BGP SR policy address family at SR-TE head-end:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure** | |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **router bgp** *as-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router bgp 65000` | Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process. |
| **Step 3** | **bgp router-id** *ip-address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp)# bgp router-id 10.1.1.1` | Configures the local router with a specified router ID. |
| **Step 4** | **address-family** {**ipv4** \| **ipv6**} **sr-policy**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 sr-policy` | Specifies either the IPv4 or IPv6 address family and enters address family configuration submode. |
| **Step 5** | **exit** | |
| **Step 6** | **neighbor** *ip-address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp)# neighbor 10.10.0.1` | Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer. |
| **Step 7** | **remote-as** *as-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 1` | Creates a neighbor and assigns a remote autonomous system number to it. |
| **Step 8** | **address-family** {**ipv4** \| **ipv6**} **sr-policy**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 sr-policy` | Specifies either the IPv4 or IPv6 address family and enters address family configuration submode. |
| **Step 9** | **route-policy** *route-policy-name* {**in** \| **out**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy pass out` | Applies the specified policy to IPv4 or IPv6 unicast routes. |

**Example: BGP SR-TE with BGPv4 Neighbor to BGP SR-TE Controller**

The following configuration shows the an SR-TE head-end with a BGPv4 session towards a BGP SR-TE controller. This BGP session is used to signal both IPv4 and IPv6 SR policies.

```
router bgp 65000
bgp router-id 10.1.1.1
 !
 address-family ipv4 sr-policy
 !
 address-family ipv6 sr-policy
 !
 neighbor 10.1.3.1
  remote-as 10
  description *** eBGP session to BGP SRTE controller ***
  address-family ipv4 sr-policy
   route-policy pass in
   route-policy pass out
  !
  address-family ipv6 sr-policy
   route-policy pass in
   route-policy pass out
  !
 !
!
```

**Example: BGP SR-TE with BGPv6 Neighbor to BGP SR-TE Controller**

The following configuration shows an SR-TE head-end with a BGPv6 session towards a BGP SR-TE controller. This BGP session is used to signal both IPv4 and IPv6 SR policies.

```
router bgp 65000
bgp router-id 10.1.1.1
 address-family ipv4 sr-policy
 !
 address-family ipv6 sr-policy
 !
 neighbor 3001::10:1:3:1
  remote-as 10
  description *** eBGP session to BGP SRTE controller ***
  address-family ipv4 sr-policy
   route-policy pass in
   route-policy pass out
  !
  address-family ipv6 sr-policy
   route-policy pass in
   route-policy pass out
  !
 !
!
```

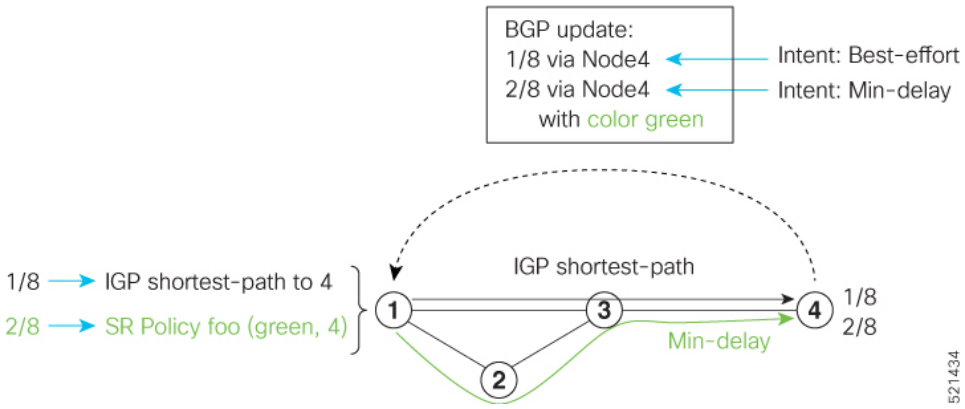
# Traffic Steering

## Automated Steering

Automated steering (AS) allows service traffic to be automatically steered onto the required transport SLA path programmed by an SR policy.

With AS, BGP automatically steers traffic onto an SR Policy based on the next-hop and color of a BGP service route. The color of a BGP service route is specified by a color extended community attribute. This color is used as a transport SLA indicator, such as min-delay or min-cost.

When the next-hop and color of a BGP service route matches the end-point and color of an SR Policy, BGP automatically installs the route resolving onto the BSID of the matching SR Policy. Recall that an SR Policy on a head-end is uniquely identified by an end-point and color.




When a BGP route has multiple extended-color communities, each with a valid SR Policy, the BGP process installs the route on the SR Policy giving preference to the color with the highest numerical value.

The granularity of AS behaviors can be applied at multiple levels, for example:

- At a service level—When traffic destined to all prefixes in a given service is associated to the same transport path type. All prefixes share the same color.
- At a destination/prefix level—When traffic destined to a prefix in a given service is associated to a specific transport path type. Each prefix could be assigned a different color.
- At a flow level—When flows destined to the same prefix are associated with different transport path types

AS behaviors apply regardless of the instantiation method of the SR policy, including:

- On-demand SR policy
- Manually provisioned SR policy
- PCE-initiated SR policy

See the Verifying BGP VRF Information, on page 171 and Verifying Forwarding (CEF) Table, on page 172 sections for sample output that shows AS implementation.

# Color-Only Automated 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).

**Note** 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 230.

### Configure Color-Only Steering

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

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


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

## 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** See Color-Only Automated Steering, on page 229 for information about color-only steering (NULL end-point).

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

| | |
|---|---|
| **co-flag 00** | **1.** The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>.<br>**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>.<br>**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.<br>**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.<br>**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)#
```

# Address-Family Agnostic Automated 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
```

# Per-Flow Automated Steering

*Table 23: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Per-Flow Automated Steering: L3 / L2 BGP Services + BSID Steering | Release 7.4.1 | This feature introduces support for BGP VPNv6 (6VPE) and BGP EVPN (single-home/multi-homed) over PFP, labeled traffic (Binding SID as top-most label in the stack) steering over per-flow policy (PFP).<br><br>An ingress QoS policy applied to an input interface is used to classify flows and set corresponding MPLS experimental values. |

The steering of traffic through a Segment Routing (SR) policy is based on the candidate paths of that policy. For a given policy, a candidate path specifies the path to be used to steer traffic to the policy's destination. The policy determines which candidate path to use based on the candidate path's preference and state. The candidate path that is valid and has the highest preference is used to steer all traffic using the given policy. This type of policy is called a Per-Destination Policy (PDP).

Per-Flow Automated Traffic Steering using SR-TE Policies introduces a way to steer traffic on an SR policy based on the attributes of the incoming packets, called a Per-Flow Policy (PFP).

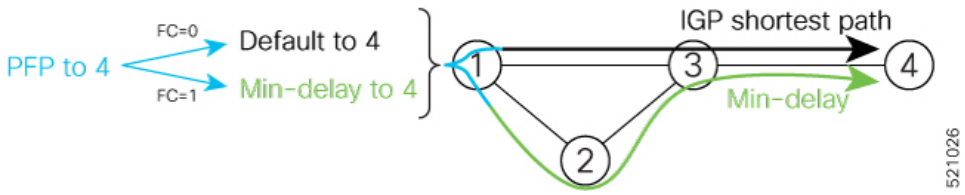
A PFP provides up to 8 "ways" or options to the endpoint. With a PFP, packets are classified by a classification policy and marked using internal tags called forward classes (FCs). The FC setting of the packet selects the "way". For example, this "way" can be a traffic-engineered SR path, using a low-delay path to the endpoint. The FC is represented as a numeral with a value of 0 to 7.

A PFP defines an array of FC-to-PDP mappings. A PFP can then be used to steer traffic into a given PDP based on the FC assigned to a packet.

As with PDPs, PFPs are identified by a {headend, color, endpoint} tuple. The color associated with a given FC corresponds to a valid PDP policy of that color and same endpoint as the parent PFP. So PFP policies contain mappings of different FCs to valid PDP policies of different colors. Every PFP has an FC designated as its default FC. The default FC is associated to packets with a FC undefined under the PFP or for packets with a FC with no valid PDP policy.

The following example shows a per-flow policy from Node1 to Node4:

*Figure 13: PFP Example*

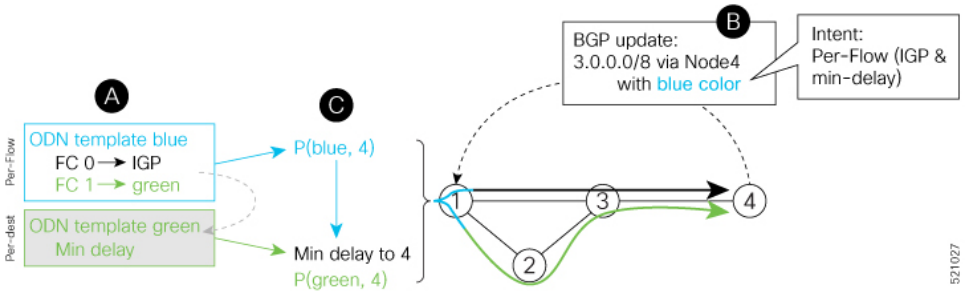



- FC=0 -> shortest path to Node4
  - IGP shortest path = 16004

- FC=1 -> Min-delay path to Node4
  - SID list = {16002,16004}

The same on-demand instantiation behaviors of PDPs apply to PFPs. For example, an edge node automatically (on demand) instantiates Per-Flow SR Policy paths to an endpoint by service route signaling. Automated Steering steers the service route in the matching SR Policy.

***Figure 14: PFP with ODN Example***




Like PDPs, PFPs have a binding SID (BSID). Existing SR-TE automated steering (AS) mechanisms for labeled traffic (via BSID) and unlabeled traffic (via BGP) onto a PFP is similar to that of a PDP. For example, a packet having the BSID of a PFP as the top label is steered onto that PFP. The classification policy on the ingress interface marks the packet with an FC based on the configured class-map. The packet is then steered to the PDP that corresponds to that FC.

### Usage Guidelines and Limitations

The following guidelines and limitations apply to the platform when acting as a head-end of a PFP policy:

- BGP IPv4 unicast over PFP (steered via ODN/AS) is supported
- BGP IPv6 unicast (with IPv4 next-hop [6PE]) over PFP (steered via ODN/AS) is supported
- BGP IPv6 unicast (with IPv6 next-hop) over PFP (steered via ODN/AS) is supported
- BGP VPNv4 over PFP is not supported
- BGP VPNv4 over PFP (steered via ODN/AS) is supported
- BGP VPNv6 (6VPE) over PFP is not supported
- BGP VPNv6 (6VPE) over PFP (steered via ODN/AS) is supported
- BGP EVPN over PFP is not supported
- BGP EVPN (single-home/multi-homed) over PFP (steered via ODN/AS) is supported
- Pseudowire and VPLS over PFP are not supported
- Pseudowire and VPLS over PFP (steered with preferred-path) are supported
- BGP multipath is not supported
- BGP multipath is supported
- BGP PIC is not supported

- Labeled traffic (Binding SID) steered over PFP is not supported
- Labeled traffic (Binding SID as top-most label in the stack) steered over PFP is supported
- When not explicitly configured, FC 0 is the default FC.
- A PFP is considered valid as long as its default FC has a valid PDP.
- An ingress QoS policy applied to an input interface is used to classify flows and set corresponding forward-class (FC) values.
- An ingress QoS policy applied to an input interface is used to classify flows and set corresponding MPLS experimental values.
- PFP implementation is accomplished with a double-pass through the ASIC (recirculation).
  - In the first pass, an ingress QoS policy applied to an input interface is used to classify flows and set MPLS EXP values, alongside push of service label and PFP Binding SID label.
  - In the absence of any ingress QoS policy, the default behavior is to copy PREC/DSCP/EXP to PFP BSID MPLS EXP.
  - In the second pass, a forwarding lookup based on PFP Binding SID label plus MPLS EXP is used to resolve to a given PFP's PDP.
- The PFP's BSID is allocated from a user-configured MPLS label block; see Configuring PFP BSID Label Block, on page 235.
- The following counters are supported:
  - PFP's BSID counter (packet, bytes)
  - Per-FC counters (packet, byte)
    - Collected from the PDP's segment-list-per-path egress counters
    - If an SR policy is used for more than one purpose (as a regular policy as well as a PDP under one or more PFPs), then the collected counters will represent the aggregate of all contributions. To preserve independent counters, it is recommended that an SR policy be used only for one purpose.
- Inbound packet classification, based on the following fields, is supported:
  - IP precedence
  - IP DSCP
  - L3 ACL-based (L3 source/destination IP; L4 source/destination port)
- A color associated with a PFP SR policy cannot be used by a non-PFP SR policy. For example, if a per-flow ODN template for color 100 is configured, then the system will reject the configuration of any non-PFP SR policy using the same color. You must assign different color value ranges for PFP and non-PFP SR policies.

### Configuring PFP BSID Label Block

Implementation on NCS platforms requires that the BSID assigned to a PFP be allocated from a preconfigured label block. The BSID is a local segment.

This label range cannot overlap with the existing SRLB or SRGB ranges allocated on the platform.

To configure the MPLS label block for PFP BSID allocation, use the **block name** *name* **type pfp start** *starting-value* {**end** *ending-value* | **size** *size* } [**client** *word*] command.

This example shows how to allocate a block of labels based on the size of the block:

```
Router(config)# mpls label blocks
Router(config-mpls-lbl-blks)# block name sample-pfp-bsid-block type pfp start 40000 size
1000 client any
```

This example shows how to allocate a block of labels based on specific starting and ending values:

```
Router(config)# mpls label blocks
Router(config-mpls-lbl-blks)# block name sample-pfp-bsid-block type pfp start 40000 end
41000 client any
```

### Configuring ODN Template for PFP Policies: Example

The following example depicts an ODN template for PFP policies that includes three FCs.

The example also includes the corresponding ODN templates for PDPs as follows:

- FC0 (default FC) mapped to color 10 = Min IGP path
- FC1 mapped to color 20 = Flex Algo 128 path
- FC2 mapped to color 30 = Flex Algo 129 path

```
segment-routing
 traffic-eng
  on-demand color 10
   dynamic
    metric
     type igp
    !
   !
  !
  on-demand color 20
   dynamic
    sid-algorithm 128
   !
  !
  on-demand color 30
   dynamic
    sid-algorithm 129
   !
  !
  on-demand color 1000
   per-flow
    forward-class 0 color 10
    forward-class 1 color 20
    forward-class 2 color 30
```

```
segment-routing
 traffic-eng
  on-demand color 10
   dynamic
    metric
     type igp
    !
   !
  !
  on-demand color 20
   constraints
    segments
     sid-algorithm 128
    !
   !
  !
  on-demand color 30
   constraints
    segments
     sid-algorithm 129
    !
   !
  !
  on-demand color 1000
   per-flow
    forward-class 0 color 10
    forward-class 1 color 20
    forward-class 2 color 30
```

### Manually Configuring a PFP and PDPs: Example

The following example depicts a manually defined PFP that includes three FCs and corresponding manually defined PDPs.

The example also includes the corresponding PDPs as follows:

- FC0 (default FC) mapped to color 10 = Min IGP path
- FC1 mapped to color 20 = Min TE path
- FC2 mapped to color 30 = Min delay path

```
mpls label blocks
 block name sample-pfp-bsid-block type pfp start 400000 size 1000 client any
 !
!
segment-routing
 traffic-eng
  policy MyPerFlow
   color 1000 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     per-flow
      forward-class 0 color 10
      forward-class 1 color 20
      forward-class 2 color 30
  !
  policy MyLowIGP
   color 10 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     dynamic
```

```
     metric type igp
  !
  policy MyLowTE
   color 20 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     dynamic
      metric type te
  !
  policy MyLowDelay
   color 30 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     dynamic
      metric type delay
```

### Configuring Ingress Classification: Example

An MQC QoS policy is used to classify and mark traffic to a corresponding fowarding class.

The following shows an example of such ingress classification policy:

```
class-map match-any MinDelay
 match dscp 46
 end-class-map
!
class-map match-any PremiumHosts
 match access-group ipv4 PrioHosts
 end-class-map
!
!
policy-map MyPerFlowClassificationPolicy
 class MinDelay
  set forward-class 2
 !
 class PremiumHosts
  set forward-class 1
 !
 class class-default
 !
 end-policy-map
!
interface GigabitEthernet0/0/0/0
 description PE_Ingress_Interface
 service-policy input MyPerFlowClassificationPolicy
!
```

### Configuring Ingress Classification: Example

An MQC QoS policy is used to classify and mark traffic to a corresponding MPLS experimental value.

The following shows an example of such ingress classification policy:

```
class-map match-any MinDelay
 match dscp 46
 end-class-map
!
class-map match-any PremiumHosts
 match access-group ipv4 PrioHosts
 end-class-map
!
!
```

```
policy-map MyPerFlowClassificationPolicy
 class MinDelay
  set mpls experimental imposition 2
 !
 class PremiumHosts
  set mpls experimental imposition 1
 !
 class class-default
 !
 end-policy-map
!
interface GigabitEthernet0/0/0/0
 description PE_Ingress_Interface
 service-policy input MyPerFlowClassificationPolicy
!
```

### Determining Per-Flow Policy State

A PFP is brought down for the following reasons:

- The PDP associated with the default FC is in a down state.
- All FCs are associated with PDPs in a down state.
- The FC assigned as the default FC is missing in the forward class mapping.

  Scenario 1—FC 0 (default FC) is not configured in the FC mappings below:

  ```
  policy foo
    color 1 end-point ipv4 10.1.1.1
    per-flow
     forward-class 1 color 10
     forward-class 2 color 20
  ```

  Scenario 2—FC 1 is configured as the default FC, however it is not present in the FC mappings:

  ```
  policy foo
    color 1 end-point ipv4 10.1.1.1
    per-flow
     forward-class 0 color 10
     forward-class 2 color 20
     forward-class default 1
  ```

# 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.

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.

## Stitching SR-TE Polices Using Binding SID: Example

In this example, three SR-TE policies are stitched together to form a seamless end-to-end path from node 1 to node 10. The path is a chain of SR-TE policies stitched together using the binding-SIDs of intermediate policies, providing a seamless end-to-end path.

***Figure 15: Stitching SR-TE Polices Using Binding SID***

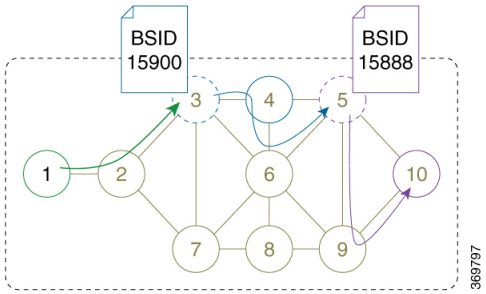



***Table 24: Router IP Address***

| Router | Prefix Address | Prefix SID/Adj-SID |
|---|---|---|
| 3 | Loopback0 - 10.1.1.3 | Prefix SID - 16003 |
| 4 | Loopback0 - 10.1.1.4<br>Link node 4 to node 6 - 10.4.6.4 | Prefix SID - 16004<br>Adjacency SID - dynamic |
| 5 | Loopback0 - 10.1.1.5 | Prefix SID - 16005 |
| 6 | Loopback0 - 10.1.1.6<br>Link node 4 to node 6 - 10.4.6.6 | Prefix SID - 16006<br>Adjacency SID - dynamic |
| 9 | Loopback0 - 10.1.1.9 | Prefix SID - 16009 |
| 10 | Loopback0 - 10.1.1.10 | Prefix SID - 16010 |

## Procedure

**Step 1** On node 5, do the following:

a) Define an SR-TE policy with an explicit path configured using the loopback interface IP addresses of node 9 and node 10.

b) Define an explicit binding-SID (**mpls label 15888**) allocated from SRLB for the SR-TE policy.

**Example:**

**Node 5**

```
segment-routing
 traffic-eng
  segment-list PATH-9_10
   index 10 address ipv4 10.1.1.9
   index 20 address ipv4 10.1.1.10
  !
  policy foo
   binding-sid mpls 15888
   color 777 end-point ipv4 10.1.1.10
   candidate-paths
    preference 100
     explicit segment-list PATH5-9_10
     !
    !
   !
  !
 !
!

RP/0/RSP0/CPU0:Node-5# show segment-routing traffic-eng policy color 777

SR-TE policy database
---------------------

Color: 777, End-point: 10.1.1.10
  Name: srte_c_777_ep_10.1.1.10
  Status:
    Admin: up  Operational: up for 00:00:52 (since Aug 19 07:40:12.662)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: foo
      Requested BSID: 15888
      PCC info:
        Symbolic name: cfg_foo_discr_100
        PLSP-ID: 70
      Explicit: segment-list PATH-9_10 (valid)
        Weight: 1, Metric Type: TE
          16009 [Prefix-SID, 10.1.1.9]
          16010 [Prefix-SID, 10.1.1.10]
  Attributes:
    Binding SID: 15888 (SRLB)
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

**Step 2** On node 3, do the following:

a) Define an SR-TE policy with an explicit path configured using the following:

- Loopback interface IP address of node 4
- Interface IP address of link between node 4 and node 6
- Loopback interface IP address of node 5
- Binding-SID of the SR-TE policy defined in Step 1 (**mpls label 15888**)

**Note**
This last segment allows the stitching of these policies.

b) Define an explicit binding-SID (**mpls label 15900**) allocated from SRLB for the SR-TE policy.

**Example:**

**Node 3**

```
segment-routing
 traffic-eng
  segment-list PATH-4_4-6_5_BSID
   index 10 address ipv4 10.1.1.4
   index 20 address ipv4 10.4.6.6
   index 30 address ipv4 10.1.1.5
   index 40 mpls label 15888
  !
  policy baa
   binding-sid mpls 15900
   color 777 end-point ipv4 10.1.1.5
   candidate-paths
    preference 100
     explicit segment-list PATH-4_4-6_5_BSID
     !
    !
   !
  !
 !
!

RP/0/RSP0/CPU0:Node-3# show segment-routing traffic-eng policy color 777

SR-TE policy database
---------------------

Color: 777, End-point: 10.1.1.5
  Name: srte_c_777_ep_10.1.1.5
  Status:
    Admin: up  Operational: up for 00:00:32 (since Aug 19 07:40:32.662)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: baa
      Requested BSID: 15900
      PCC info:
        Symbolic name: cfg_baa_discr_100
        PLSP-ID: 70
      Explicit: segment-list PATH-4_4-6_5_BSID (valid)
        Weight: 1, Metric Type: TE
          16004 [Prefix-SID, 10.1.1.4]
          80005 [Adjacency-SID, 10.4.6.4 - 10.4.6.6]
          16005 [Prefix-SID, 10.1.1.5]
          15888
  Attributes:
    Binding SID: 15900 (SRLB)
    Forward Class: 0
    Steering BGP disabled: no
```

```
        IPv6 caps enable: yes
```

**Step 3** On node 1, define an SR-TE policy with an explicit path configured using the loopback interface IP address of node 3 and the binding-SID of the SR-TE policy defined in step 2 (**mpls label 15900**). This last segment allows the stitching of these policies.

**Example:**

**Node 1**

```
segment-routing
 traffic-eng
  segment-list PATH-3_BSID
   index 10 address ipv4 10.1.1.3
   index 20 mpls label 15900
  !
  policy bar
   color 777 end-point ipv4 10.1.1.3
   candidate-paths
    preference 100
     explicit segment-list PATH-3_BSID
     !
    !
   !
  !
 !
!

RP/0/RSP0/CPU0:Node-1# show segment-routing traffic-eng policy color 777

SR-TE policy database
---------------------

Color: 777, End-point: 10.1.1.3
  Name: srte_c_777_ep_10.1.1.3
  Status:
    Admin: up  Operational: up for 00:00:12 (since Aug 19 07:40:52.662)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: bar
      Requested BSID: dynamic
      PCC info:
        Symbolic name: cfg_bar_discr_100
        PLSP-ID: 70
      Explicit: segment-list PATH-3_BSID (valid)
        Weight: 1, Metric Type: TE
          16003 [Prefix-SID, 10.1.1.3]
          15900
  Attributes:
    Binding SID: 80021
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

# L2VPN Preferred Path

EVPN VPWS Preferred Path over SR-TE Policy feature allows you to set the preferred path between the two end-points for EVPN VPWS pseudowire (PW) using SR-TE policy.

L2VPN VPLS or VPWS Preferred Path over SR-TE Policy feature allows you to set the preferred path between the two end-points for L2VPN Virtual Private LAN Service (VPLS) or Virtual Private Wire Service (VPWS) using SR-TE policy.

# Static Route over Segment Routing Policy

This feature allows you to specify a Segment Routing (SR) policy as an interface type when configuring static routes for MPLS data planes.

For information on configuring static routes, see the "Implementing Static Routes" chapter in the *Routing Configuration Guide for Cisco NCS 540 Series Routers*.

### Configuration Example

The following example depicts a configuration of a static route for an IPv4 destination over an SR policy.

```
Router(config)# router static
Router(config-static)# address-family ipv4 unicast
Router(config-static-afi)# 10.1.100.100/32 sr-policy sample-policy
```

### Running Configuration

```
Router# show run segment-routing traffic-eng

segment-routing
 traffic-eng
  segment-list sample-SL
   index 10 mpls adjacency 10.1.1.102
   index 20 mpls adjacency 10.1.1.103
  !
  policy sample-policy
   color 777 end-point ipv4 10.1.1.103
   candidate-paths
    preference 100
     explicit segment-list sample-SL

Router# show run segment-routing traffic-eng

router static
 address-family ipv4 unicast
  10.1.1.4/32 sr-policy srte_c_200_ep_10.1.1.4
 !
!
```

### Verification

```
Router# show segment-routing traffic-eng policy candidate-path name sample-policy

SR-TE policy database
---------------------

Color: 777, End-point: 10.1.1.103
  Name: srte_c_777_ep_10.1.1.103
  Status:
    Admin: up  Operational: up for 00:06:35 (since Jan 17 14:34:35.120)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: sample-policy
```

```
      Requested BSID: dynamic
      PCC info:
        Symbolic name: cfg_sample-policy_discr_100
        PLSP-ID: 5
      Constraints:
        Protection Type: protected-preferred
        Maximum SID Depth: 9
      Explicit: segment-list sample-SL (valid)
        Weight: 1, Metric Type: TE
          SID[0]: 100102 [Prefix-SID, 10.1.1.102]
          SID[1]: 100103 [Prefix-SID, 10.1.1.103]
  Attributes:
    Binding SID: 24006
    Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
    Max Install Standby Candidate Paths: 0
```

```
Router# show static sr-policy sample-policy

SR-Policy-Name           State   Binding-label Interface                 ifhandle   VRF
       Paths
sample-policy            Up      24006         srte_c_777_ep_10.1.1.103  0x2000803c default
      10.1.100.100/32
Reference count=1, Internal flags=0x0
Last Policy notification was Up at Jan 17 13:39:46.478
```

```
Router# show route 10.1.100.100/32

Routing entry for 10.1.100.100/32
  Known via "static", distance 1, metric 0
  Installed Jan 17 14:35:40.969 for 00:06:38
  Routing Descriptor Blocks
    directly connected, via srte_c_777_ep_10.1.1.103
      Route metric is 0
  No advertising protos.
```

```
Router# show route 10.1.100.100/32 detail

Routing entry for 10.1.100.100/32
  Known via "static", distance 1, metric 0
  Installed Jan 17 14:35:40.969 for 00:06:44
  Routing Descriptor Blocks
    directly connected, via srte_c_777_ep_10.1.1.103
      Route metric is 0
      Label: None
      Tunnel ID: None
      Binding Label: 0x5dc6 (24006)
      Extended communities count: 0
      NHID: 0x0 (Ref: 0)
  Route version is 0x1 (1)
  No local label
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_STATIC (9) SVD Type RIB_SVD_TYPE_LOCAL
  Download Priority 3, Download Version 3169
  No advertising protos.
```

```
Router# show cef 10.1.100.100/32

10.1.100.100/32, version 3169, internal 0x1000001 0x30 (ptr 0x8b1b95d8) [1], 0x0 (0x0), 0x0
 (0x0)
 Updated Jan 17 14:35:40.971
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
  gateway array (0x8a92f228) reference count 1, flags 0x2010, source rib (7), 0 backups
                [1 type 3 flags 0x48441 (0x8a9d1b68) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Jan 17 14:35:40.971
 LDI Update time Jan 17 14:35:40.972
   via local-label 24006, 3 dependencies, recursive [flags 0x0]
    path-idx 0 NHID 0x0 [0x8ac59f30 0x0]
    recursion-via-label
    next hop via 24006/1/21

    Load distribution: 0 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   recursive                 24006/1
```

# Autoroute Include

***Table 25: Feature History Table***

| Feature Name | Release | Description |
|---|---|---|
| Autoroute Include | Release 7.3.2 | This feature allows you to steer specific IGP (IS-IS, OSPF) prefixes, or all prefixes, over non-shortest paths and to divert the traffic for those prefixes on to an SR-TE policy. |

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

The **autoroute include all** option applies Autoroute Announce functionality for all destinations or prefixes.

The **autoroute include ipv4** *address* option applies Autoroute Destination functionality for the specified destinations or prefixes. This option is supported for IS-IS only; it is not supported for OSPF.

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.

### Usage Guidelines and Limitations

- Autoroute Include supports three metric types:
  - Default (no metric): The path over the SR-TE policy inherits the shortest path metric.
  - Absolute (constant) 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. Use the **autoroute metric constant** *constant-metric* command, where *constant-metric* is from 1 to 2147483647.

- Relative metric: The shortest path metric to the policy endpoint is modified with the relative value configured (plus or minus). Use the **autoroute metric relative** *relative-metric* command, where *relative-metric* is from -10 to +10.

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

- LDP over SR-TE not supported.
- LDP to SR-TE interworking is not supported.
- Static route over SR-TE is not supported.

### Configuration Examples

The following example shows how to configure autoroute include for all prefixes:

```
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-point ipv4 10.1.1.2
Router(config-sr-te-policy)# autoroute include all
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
```

The following example shows how to configure autoroute include for the specified IPv4 prefixes:

**Note** This option is supported for IS-IS only; it is not supported for OSPF.

```
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-point ipv4 10.1.1.2
Router(config-sr-te-policy)# autoroute include ipv4 10.1.1.21/32
Router(config-sr-te-policy)# autoroute include ipv4 10.1.1.23/32
Router(config-sr-te-policy)# autoroute metric constant 1
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-pp-index)# explicit segment-list Plist-1
```

# Policy-Based Tunnel Selection for SR-TE Policy

Policy-Based Tunnel Selection (PBTS) is a mechanism that lets you direct traffic into specific SR-TE policies based on different classification criteria. PBTS benefits Internet service providers (ISPs) that carry voice and data traffic through their networks, who want to route this traffic to provide optimized voice service.

PBTS works by selecting SR-TE policies based on the classification criteria of the incoming packets, which are based on the IP precedence, experimental (EXP), differentiated services code point (DSCP), or type of service (ToS) field in the packet. Default-class configured for paths is always zero (0). If there is no TE for a given forward-class, then the default-class (0) will be tried. If there is no default-class, then the packet is dropped. PBTS supports up to seven (exp 1 - 7) EXP values associated with a single SR-TE policy.

For more information about PBTS, refer to the "Policy-Based Tunnel Selection" section in the *MPLS Configuration Guide for Cisco NCS 540 Series RoutersMPLS Configuration Guide*.

### Configure Policy-Based Tunnel Selection for SR-TE Policies

The following section lists the steps to configure PBTS for an SR-TE policy.

**Note** Steps 1 through 4 are detailed in the "Implementing MPLS Traffic Engineering" chapter of the *MPLS Configuration Guide for Cisco NCS 540 Series RoutersMPLS Configuration Guide*.

1. Define a class-map based on a classification criteria.
2. Define a policy-map by creating rules for the classified traffic.
3. Associate a forward-class to each type of ingress traffic.
4. Enable PBTS on the ingress interface, by applying this service-policy.
5. Create one or more egress SR-TE policies (to carry packets based on priority) to the destination and associate the egress SR-TE policy to a forward-class.

### Configuration Example

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy POLICY-PBTS
Router(config-sr-te-policy)# color 1001 end-point ipv4 10.1.1.20
Router(config-sr-te-policy)# autoroute
Router(config-sr-te-policy-autoroute)# include all
Router(config-sr-te-policy-autoroute)# forward-class 1
Router(config-sr-te-policy-autoroute)# exit
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 1
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST1
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# preference 2
Router(config-sr-te-policy-path-pref)# dynamic
Router(config-sr-te-pp-info)# metric
Router(config-sr-te-path-metric)# type te
Router(config-sr-te-path-metric)# commit
```

### Running Configuration

```
segment-routing
 traffic-eng
  policy POLICY-PBTS
   color 1001 end-point ipv4 10.1.1.20
   autoroute
    include all
    forward-class 1
   !
   candidate-paths
    preference 1
     explicit segment-list SIDLIST1
     !
    !
    preference 2
     dynamic
      metric
       type te
```

# Miscellaneous

## SR Policy Liveness Monitoring

SR Policy liveness monitoring allows you to verify end-to-end traffic forwarding over an SR Policy candidate path by periodically sending performance monitoring (PM) packets. The head-end router sends PM packets to the SR policy's endpoint router, which sends them back to the head-end without any control-plane dependency on the endpoint router.

For more information about this feature, see SR Policy Liveness Monitoring, on page 327.

# Programming Non-Active Candidate Paths of an SR Policy

*Table 26: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Programming Non-Active Candidate Paths of an SR Policy | Release 7.6.1 | By programming non-active candidate paths (CPs) in the forwarding plane, you ensure that if the existing active CP is unavailable, the traffic switches quickly to the new CP, thus minimizing loss of traffic flow.<br>In earlier releases, instantiating a non-active CP to the forwarding plane after the unavailability of the active CP could take a few seconds, resulting in potential loss of traffic flow.<br>This feature introduces the following command:<br>• max-install-standby-cpaths |

An SR Policy is associated with one or more candidate paths (CP). A CP is selected as the active CP when it is valid and it has the highest preference value among all the valid CPs of the SR Policy. By default, only the active CP is programmed in the forwarding plane.

This feature allows the programming of multiple CPs of an SR policy in the forwarding plane. This minimizes traffic loss when a new CP is selected as active.

### Usage Guidelines and Limitations

Observe the following usage guidelines and limitations:

- Up to three non-active CPs can be programmed in the forwarding plane.
- Manually configured CPs are supported. This includes CPs with explicit paths or dynamic (head-end computed or PCE-delegated) paths.
- On-Demand instantiated CPs (ODN) are supported.
- BGP-initiated CPs are supported.
- PCE-initiated CPs via PCEP are not supported. This applies to polices created via CLI or via north-bound HTTP-based API.
- Programming of non-active CPs is not supported with SRv6-TE policies, Per-Flow Policies (PFP), or point-to-multipoint SR policies (Tree-SID)
- PCEP reporting of additional CPs is supported, but the PCEP reporting does not distinguish between active and non-active CPs.

- Programming of non-active CPs can be enabled for all SR policies (global), for a specific policy (local), or ODN template.

  If enabled globally and also locally or on ODN template, the local or ODN configuration takes precedence over the global configuration.

  - Programming of non-active CPs under global SR-TE and configuring policy path protection of an SR policy is supported. In this case, policy path protection takes precedence.
  - Programming of non-active CPs for a specific SR policy and configuring policy path protection of an SR policy is not supported.

- The number of policies supported could be impacted by the number of non-active CPs per policy. Programming non-active CPs in the forwarding plane consumes hardware resources (such as local label and ECMP FEC) when more candidate paths are pre-programmed in forwarding than are actually carrying traffic.
- The active CP will be in programmed state. The remaining CPs will be in standby programmed state.
- We recommend that you create separate PM sessions for active and standby candidate paths to monitor the health of the paths end-to-end.

  The recommended PM timers should be different for active and standby PM profiles. The PM timers should be less aggressive for the standby PM profile compared to the active PM profile. See Configure Performance Measurement, on page 321 for information about configuring PM sessions.

**Note** PM sessions for BGP-TE policies are not supported. PM profiles can be configured only under configured policies at the head-end.

- The protected paths for each CP is programmed in the respective LSPs. The protected paths of active CPs are programmed in the active LSP, and the protected paths of standby CPs are programmed in the standby LSP.
- If a candidate path with higher preference becomes available, the traffic will switch to it in Make-Before-Break (MBB) behavior.

### Configuration

Programming of non-active CPs can be enabled for all SR policies (global), for a specific policy (local), or ODN template. If enabled globally, the local or ODN configuration takes precedence over the global configuration.

#### Global SR-TE

Use the **max-install-standby-cpaths** *value* command to configure standby candidate paths for all SR policies, for a specific policy, or for an ODN template. The range for *value* is from 1 to 3. Use **no max-install-standby-cpaths** command to return to the default behavior.

The following example shows how to configure standby candidate paths globally:

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# max-install-standby-cpaths 2
Router(config-sr-te)#
```

### Running Config

```
segment-routing
 traffic-eng
  max-install-standby-cpaths 2
```

### Local SR Policy

Use the **max-install-standby-cpaths** *value* command to configure standby candidate paths for a specific policy. The range for *value* is from 0 (disable) to 3.

If programming of non-active CPs is enabled for all SR policies (global), you can disable programming of non-active CPs for a specific policy using the **max-install-standby-cpaths 0** command.

The following example shows how to configure standby candidate paths for a specific SR policy:

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy MyBackupPolicy
Router(config-sr-te-policy)# max-install-standby-cpaths 2
Router(config-sr-te-policy)#
```

### Running Config

```
segment-routing
 traffic-eng
  policy MyBackupPolicy
   max-install-standby-cpaths 2
```

### SR ODN

When you create an ODN template, two CPs are created by default (PCE-delegated and head-end computed) with preference 100 and preference 200. You can use the **max-install-standby-cpaths 1** command to program the non-active CP in forwarding. If programming of non-active CPs is enabled for all SR policies (global), you can disable programming of non-active CPs on ODN template using the **max-install-standby-cpaths 0** command.

The following example shows how to configure standby candidate paths for an SR ODN template:

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color 10
Router(config-sr-te-color)# max-install-standby-cpaths 1
Router(config-sr-te-color)#
```

Running Config

```
segment-routing
 traffic-eng
   on-demand color 10
    max-install-standby-cpaths 1
```

The following example shows how to enable three standby CPs globally and disable standby CPs on local SR policy and ODN template:

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# max-install-standby-cpaths 3
Router(config-sr-te)# policy MyBackupPolicy
Router(config-sr-te-policy)# max-install-standby-cpaths 0
```

```
Router(config-sr-te-policy)# exit
Router(config-sr-te)# on-demand color 10
Router(config-sr-te-color)# max-install-standby-cpaths 0
Router(config-sr-te-color)#
```

### Verification

The following output shows the status of active and backup CPs:

```
Router# show segment-routing traffic-eng policy

SR-TE policy database
---------------------

Color: 50, End-point: 1.1.1.4
  Name: srte_c_50_ep_1.1.1.4
  Status:
    Admin: up  Operational: up for 08:17:32 (since Sep  9 13:16:02.818)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_100
        PLSP-ID: 2
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      Explicit: segment-list WORKING (valid)
        Reverse: segment-list REVERSE_WORKING
        Weight: 1, Metric Type: TE
          24010
          24012
    Preference: 80 (configuration) (standby)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_80
        PLSP-ID: 3
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      Explicit: segment-list STANDBY1 (valid)
        Reverse: segment-list REVERSE_STANDBY1
        Weight: 1, Metric Type: TE
          24018
          24010
    Preference: 60 (configuration) (standby)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_60
        PLSP-ID: 4
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      Explicit: segment-list STANDBY2 (valid)
        Reverse: segment-list REVERSE_STANDBY2
        Weight: 1, Metric Type: TE
          24014
    Preference: 40 (configuration)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_40
        PLSP-ID: 5
        Protection Type: protected-preferred
```

```
        Maximum SID Depth: 10
      Dynamic (valid)
        Metric Type: TE,   Path Accumulated Metric: 10
          24005 [Adjacency-SID, 13.13.13.1 - 13.13.13.4]
    Preference: 30 (configuration)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_30
        PLSP-ID: 1
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      Dynamic (pce 1.1.1.4) (valid)
        Metric Type: TE,   Path Accumulated Metric: 10
          24015 [Adjacency-SID, 13.13.13.1 - 13.13.13.4]
    Preference: 20 (configuration)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_20
        PLSP-ID: 6
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      Explicit: segment-list WORKING2 (valid)
        Reverse: segment-list REVERSE_WORKING2
        Weight: 1, Metric Type: TE
          24012
          24012
    Preference: 10 (configuration)
      Name: NCP_STATIC
      Requested BSID: 5000
      PCC info:
        Symbolic name: cfg_NCP_STATIC_discr_10
        PLSP-ID: 7
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      Explicit: segment-list WORKING3 (valid)
        Reverse: segment-list REVERSE_WORKING3
        Weight: 1, Metric Type: TE
          24010
          24014
          24010
  Attributes:
    Binding SID: 5000
    Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
    Max Install Standby CPaths: 2


Router# show segment-routing traffic-eng forwarding policy

SR-TE Policy Forwarding database
--------------------------------

Color: 50, End-point: 1.1.1.4
  Name: srte_c_50_ep_1.1.1.4
  Binding SID: 5000
  Active LSP:
    Candidate path:
      Preference: 100 (configuration)
      Name: NCP_STATIC
```

```
  Local label: 24021
  Segment lists:
    SL[0]:
      Name: WORKING
      Switched Packets/Bytes: 0/0
      Paths:
        Path[0]:
          Outgoing Label: 24012
          Outgoing Interfaces: GigabitEthernet0/0/0/0
          Next Hop: 10.10.10.2
          Switched Packets/Bytes: 0/0
          FRR Pure Backup: No
          ECMP/LFA Backup: No
          Internal Recursive Label: Unlabelled (recursive)
          Label Stack (Top -> Bottom): { 24012 }
Standby LSP(s):
  LSP[0]:
    Candidate path:
      Preference: 80 (configuration)
      Name: NCP_STATIC
    Local label: 24024
    Segment lists:
      SL[0]:
        Name: STANDBY1
        Switched Packets/Bytes: 0/0
        Paths:
          Path[0]:
            Outgoing Label: 24010
            Outgoing Interfaces: GigabitEthernet0/0/0/2
            Next Hop: 12.12.12.3
            Switched Packets/Bytes: 0/0
            FRR Pure Backup: No
            ECMP/LFA Backup: No
            Internal Recursive Label: Unlabelled (recursive)
            Label Stack (Top -> Bottom): { 24010 }
  LSP[1]:
    Candidate path:
      Preference: 60 (configuration)
      Name: NCP_STATIC
    Local label: 24025
    Segment lists:
      SL[0]:
        Name: STANDBY2
        Switched Packets/Bytes: 0/0
        Paths:
          Path[0]:
            Outgoing Label: Pop
            Outgoing Interfaces: GigabitEthernet0/0/0/3
            Next Hop: 13.13.13.4
            Switched Packets/Bytes: 0/0
            FRR Pure Backup: No
            ECMP/LFA Backup: No
            Internal Recursive Label: Unlabelled (recursive)
            Label Stack (Top -> Bottom): { Pop }

Policy Packets/Bytes Switched: 2/136
```

# 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*.

For more information about Autoroute, see the *Autoroute Announce for SR-TE* section.

**Note** 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_10.1.1.2**

### 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
Router(config-sr-te)# segment-list sample-sid-list
Router(config-sr-te-sl)# index 10 address ipv4 10.1.1.7
Router(config-sr-te-sl)# index 20 address ipv4 10.1.1.2
Router(config-sr-te-sl)# exit
Router(config-sr-te)# policy sample_policy
Router(config-sr-te-policy)# color 1000 end-point ipv4 10.1.1.2
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_10.1.1.2 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 10.1.1.7
   index 20 address ipv4 10.1.1.2
  !
  policy sample_policy
   color 1000 end-point ipv4 10.1.1.2
   candidate-paths
    preference 100
     explicit segment-list sample-sid-list
     !
    !
   !
  !
 !
!

mpls ldp
 address-family ipv4
  neighbor sr-policy srte_c_1000_ep_10.1.1.2 targeted
  discovery targeted-hello accept
 !
!
```

### Verification

```
Router# show mpls ldp interface brief
Interface       VRF Name            Config Enabled IGP-Auto-Cfg TE-Mesh-Grp cfg
--------------- ------------------- ------ ------- ------------ ---------------
Te0/3/0/0/3     default             Y      Y       0            N/A
Te0/3/0/0/6     default             Y      Y       0            N/A
Te0/3/0/0/7     default             Y      Y       0            N/A
Te0/3/0/0/8     default             N      N       0            N/A
Te0/3/0/0/9     default             N      N       0            N/A
srte_c_1000_    default             Y      Y       0            N/A


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_10.1.1.2 (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: 10.1.1.2
  Name: srte_c_1000_ep_10.1.1.2
  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: TE
          16007 [Prefix-SID, 10.1.1.7]
          16002 [Prefix-SID, 10.1.1.2]
  Attributes:
    Binding SID: 80011
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes


Router# show mpls ldp neighbor 10.1.1.2 detail

Peer LDP Identifier: 10.1.1.2:0
  TCP connection: 10.1.1.2:646 - 10.1.1.6:57473
  Graceful Restart: No
  Session Holdtime: 180 sec
  State: Oper; Msgs sent/rcvd: 421/423; Downstream-Unsolicited
  Up time: 05:22:02
  LDP Discovery Sources:
    IPv4: (1)
      Targeted Hello (10.1.1.6 -> 10.1.1.2, active/passive)
    IPv6: (0)
  Addresses bound to this peer:
    IPv4: (9)
      10.1.1.2       2.2.2.99       10.1.2.2       10.2.3.2
      10.2.4.2       10.2.22.2      10.2.222.2     10.30.110.132
      11.2.9.2
    IPv6: (0)
  Peer holdtime: 180 sec; KA interval: 60 sec; Peer state: Estab
```

```
  NSR: Disabled
  Clients: LDP over SR Policy
  Capabilities:
    Sent:
      0x508  (MP: Point-to-Multipoint (P2MP))
      0x509  (MP: Multipoint-to-Multipoint (MP2MP))
      0x50a  (MP: Make-Before-Break (MBB))
      0x50b  (Typed Wildcard FEC)
    Received:
      0x508  (MP: Point-to-Multipoint (P2MP))
      0x509  (MP: Multipoint-to-Multipoint (MP2MP))
      0x50a  (MP: Make-Before-Break (MBB))
      0x50b  (Typed Wildcard FEC)
```

# SR-TE MPLS Label Imposition Enhancement

The SR-TE MPLS Label Imposition Enhancement feature increases the maximum label imposition capabilities of the platform.

In previous releases, the platform supported:

- Up to 5 MPLS transport labels when no MPLS service labels are imposed
- Up to 3 MPLS transport labels when MPLS service labels are imposed

With the SR-TE MPLS Label Imposition Enhancement feature, the platform supports the following:

- Up to 12 MPLS transport labels when no MPLS service labels are imposed
- Up to 9 MPLS transport labels when MPLS service labels are imposed

This enhancement is enabled and disabled dynamically, as the label count changes. For example, if a path requires only 3 MPLS transport labels, the MPLS Label Imposition Enhancement feature is not enabled.

You can disable labeled services for SR-TE policies. The label switching database (LSD) needs to know if labeled services are disabled on top of an SR-TE policy to perform proper label stack splitting.

### Disable Labeled Services per Local Policy

Use the **labeled-services disable** command to disable steering for labeled services for a configured policy. This configuration applies per policy.

```
segment-routing
  traffic-eng
    policy policy name
      steering
        labeled-services disable
```

### Disable Labeled Services per ODN color

Use the **labeled-services disable** command to disable steering of labeled-services for on-demand color policies. This configuration applies for a specific ODN color.

```
segment-routing
  traffic-eng
    on-demand color color
      steering
        labeled-services disable
```

### Disable Labeled Services per Policy Type

Use the **labeled-services disable** command to disable steering of labeled services for all policies for the following policy types:

- **all** — all policies
- **local** — all locally configured policies
- **on-demand** — all BGP on-demand color policies
- **bgp-srte** — all controller-initiated BGP SR-TE policies
- **pcep** — all PCE-initiated policies

**Note** You can specify more than one policy type.

```
segment-routing
  traffic-eng
    steering
      labeled-services
         disable {all | local | on-demand | bgp-srte | pcep}
```

### Verification

Use the **show segment-routing traffic-eng policy** command to display SR policy information. The following output shows that steering of labeled services for the on-demand SR policy are disabled.

```
Router# show segment-routing traffic-eng policy color 10
Thu Jul 18 11:35:25.124 PDT

SR-TE policy database
---------------------

Color: 10, End-point: 10.1.1.8
  Name: srte_c_10_ep_10.1.1.8
  Status:
    Admin: up  Operational: up for 00:00:06 (since Jul 18 11:35:19.350)
  Candidate-paths:
    Preference: 1 (configuration) (active)
      Name: test_pol_2
      Requested BSID: dynamic
      Dynamic (valid)
        Metric Type: TE,   Path Accumulated Metric: 10
          24004 [Adjacency-SID, 10.1.1.1 - 10.1.1.2]
  Attributes:
    Binding SID: 24011
    Forward Class: 0
    Steering labeled-services disabled: yes
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

# Path invalidation drop

*Table 27: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Path invalidation drop | Release 7.4.1 | By default, if an SR Policy becomes invalid (for example, if there is no valid candidate path available), traffic falls back to the native SR forwarding path. In some scenarios, a network operator may require that certain traffic be only carried over the path associated with an SR policy and never allow the native SR LSP to be used.<br>This feature allows the SR policy to stay up in the control plane (to prevent prefixes mapped to the SR policy from falling back to the native SR LSP) but drop the traffic sent on the SR policy. |

By default, if an SR Policy becomes invalid, traffic would fall back to the native SR forwarding path.

In some scenarios, a network operator may require that certain traffic be only carried over the path associated with an SR policy and never allow the native SR LSP to be used. The SR-TE path invalidation drop feature is introduced to meet this requirement.

With the path invalidation drop feature enabled, an SR policy that would become invalid (with no existing valid candidate path) is programmed to drop traffic. At the same time, the SR policy remains operationally UP to prevent prefixes steered over this SR policy from falling back to the native SR path.

Forwarding over the SR policy path resumes without dropping traffic, when the SR policy path becomes valid.

**Note** This feature applies to an SR policy that transitions from valid to invalid; it does not apply to an SR policy that has never been declared valid.

### Enable path invalidation drop for manual SR policy

Use the **segment-routing traffic-eng policy** *name* **steering path-invalidation drop** command to enable the dropping of traffic when all candidate paths of an SR Policy becomes invalid.

```
segment-routing
 traffic-eng
  policy foo
   steering
    path-invalidation drop
```

### Enable path Invalidation Drop for On-Demand SR Policy

Use the **segment-routing traffic-eng on-demand color** *color* **steering path-invalidation drop** command (where *color* is from 1 to 4294967295) to enable the dropping of traffic when an On-Demand SR Policy becomes invalid.

```
segment-routing
 traffic-eng
  on-demand color 10
    steering
      path-invalidation drop
```

### Enable Path invalidation drop for PCE-initiated SR policy

Use the **segment-routing traffic-eng pcc profile** *profile* **steering path-invalidation drop** command (where *profile* is from 1 to 65534) to enable the dropping of traffic when a PCE-Initiated SR Policy becomes invalid.

```
segment-routing
 traffic-eng
  pcc
   profile 7
    steering
     path-invalidation drop
```

### SR Policy output with path invalidation drop enabled and active

This output shows a detailed SR policy with path invalidation drop enabled and valid candidate path:

```
Router# show segment-routing traffic-eng policy detail

SR-TE policy database
---------------------
Color: 1, End-point: 10.10.10.1
  Name: srte_c_1_ep_10.10.10.1
  Status:
    Admin: up Operational: up (path-invalidation drop) for 00:30:58 (since Feb 11
09:07:21.549)
  Candidate-paths:
     Preference: 1 (configuration) (inactive)
       Name: test
       Requested BSID: dynamic
       PCC info:
         Symbolic name: cfg_test_discr_1
         PLSP-ID: 1
         Invalidation drop enabled: yes
         Constraints:
           Protection Type: protected-preferred
           Maximum SID Depth: 10
         Performance-measurement:
           Reverse-path segment-list:
           Delay-measurement: Disabled
           Liveness-detection: Disabled
         Explicit: segment-list sl1 (inactive)
         Last error: unresolved first label (12345)
           Weight: 1, Metric Type: TE (IANA PCEP/IGP: 2/2)
             SID[0]: 12345
     Preference: 0 (SR-TE) (active) (drop)
       Name: srte_c_1_ep_10.10.10.1_discr_0
       Requested BSID: dynamic
     LSPs:
       LSP[0]:
```

```
    LSP-ID: 4 policy ID: 1 (active)
    State: Programmed
    Binding SID: 24003
Attributes:
  Binding SID: 24003
  Forward Class: Not Configured
  Steering labeled-services disabled: no
  Steering BGP disabled: no
  IPv6 caps enable: no
  Invalidation drop enabled: yes
  Max Install Standby Candidate Paths: 0
  Path Type: SRMPLSv4
```

### Configure path invalidation drop with performance measurement liveness detection

The Path Invalidation Drop feature can work alongside the **invalidation-action down** configuration in the Performance Measurement Liveness Detection feature. The Performance Measurement Liveness Detection feature enables end-to-end SR policy liveness detection for all segment lists of the active and standby candidate paths that are in the forwarding table. When **invalidation-action down** is configured and a candidate path becomes invalid, the candidate path is immediately operationally brought down and becomes invalid.

See SR Policy Liveness Monitoring, on page 327 for information about configuring liveness detection and the invalidation action.

When both **path-invalidation drop** and **performance-measurement liveness-detection invalidation-action down** are enabled, the following behavior is observed:

1. If the PM liveness session goes down, the current active candidate path of the SR policy becomes invalid.
2. SR-TE path re-optimization occurs immediately to find the next valid candidate path in the SR policy.
3. If there is no valid candidate path in the SR policy, the policy remains operationally UP, but is marked with the invalidation-drop state.

# SR-TE Reoptimization Timers

SR-TE path re-optimization occurs when the head-end determines that there is a more optimal path available than the one currently used. For example, in case of a failure along the SR-TE LSP path, the head-end could detect and revert to a more optimal path by triggering re-optimization.

Re-optimization can occur due to the following events:

- The explicit path hops used by the primary SR-TE LSP explicit path are modified
- The head-end determines the currently used path-option are invalid due to either a topology path disconnect, or a missing SID in the SID database that is specified in the explicit-path
- A more favorable path-option (lower index) becomes available

For event-based re-optimization, you can specify various delay timers for path re-optimization. For example, you can specify how long to wait before switching to a reoptimized path

Additionally, you can configure a timer to specify how often to perform reoptimization of policies. You can also trigger an immediate reoptimization for a specific policy or for all policies.

### SR-TE Reoptimization

To trigger an immediate SR-TE reoptimization, use the **segment-routing traffic-eng reoptimization** command in Exec mode:

```
Router# segment-routing traffic-eng reoptimization {all | name  policy}
```

Use the **all** option to trigger an immediate reoptimization for all policies. Use the **name** *policy* option to trigger an immediate reoptimization for a specific policy.

### Configuring SR-TE Reoptimization Timers

Use these commands in SR-TE configuration mode to configure SR-TE reoptimization timers:

- **timers candidate-path cleanup-delay** *seconds*—Specifies the delay before cleaning up candidate paths, in seconds. The range is from 0 (immediate clean-up) to 86400; the default value is 120
- **timers cleanup-delay** *seconds*—Specifies the delay before cleaning up previous path, in seconds. The range is from 0 (immediate clean-up) to 300; the default value is 10.
- **timers init-verify-restart** *seconds* —Specifies the delay for topology convergence after the topology starts populating due to a restart, in seconds. The range is from 10 to 10000; the default is 40.
- **timers init-verify-startup** *seconds*—Specifies the delay for topology convergence after topology starts populating for due to startup, in seconds. The range is from 10 to 10000; the default is 300
- **timers init-verify-switchover** *seconds*—Specifies the delay for topology convergence after topology starts populating due to a switchover, in seconds. The range is from 10 to 10000; the default is 60.
- **timers install-delay** *seconds*—Specifies the delay before switching to a reoptimized path, in seconds. The range is from 0 (immediate installation of new path) to 300; the default is 10.
- **timers periodic-reoptimization** *seconds*—Specifies how often to perform periodic reoptimization of policies, in seconds. The range is from 0 to 86400; the default is 600.

### Example Configuration

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# timers
Router(config-sr-te-timers)# candidate-path cleanup-delay 600
Router(config-sr-te-timers)# cleanup-delay 60
Router(config-sr-te-timers)# init-verify-restart 120
Router(config-sr-te-timers)# init-verify-startup 600
Router(config-sr-te-timers)# init-verify-switchover 30
Router(config-sr-te-timers)# install-delay 60
Router(config-sr-te-timers)# periodic-reoptimization 3000
```

### Running Config

```
segment-routing
 traffic-eng
  timers
   install-delay 60
   periodic-reoptimization 3000
   cleanup-delay 60
   candidate-path cleanup-delay 600
   init-verify-restart 120
   init-verify-startup 600
   init-verify-switchover 30
```

```
  !
 !
!
```

# SRv6 policy counters POL.CP.SL.INT.E

Policy counters are metrics used in networking to measure and report traffic statistics for specific policies. These counters help network administrators to monitor and manage network performance, capacity planning, and traffic engineering.

***Table 28: Feature History Table***

**POL.CP.SL.INT.E** stands for **Per-SR-policy, per Candidate-Path, per-Segment-List, per-interface, egress traffic counter**. This counter in SRv6 is used to measure and report traffic statistics for specific paths and interfaces within a network policy. It provides granular insights into the traffic flow through different segments and interfaces of a network policy, enabling effective network management and optimization.

### Benefits of SRv6 policy counters POL.CP.SL.INT.E

The benefits of SRv6 policy counters POL.CP.SL.INT.E, are as listed.

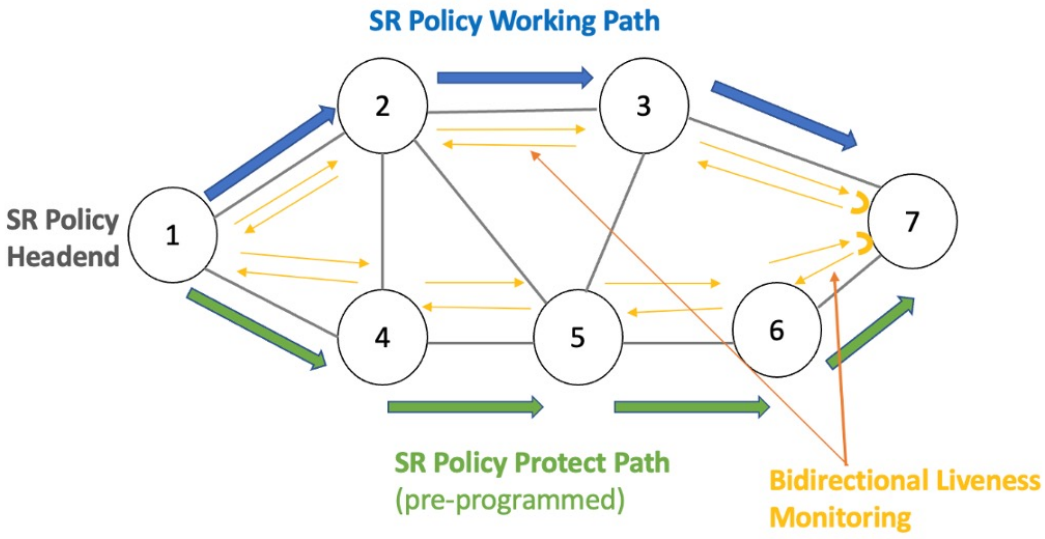
- **Network Optimization**: By analyzing the detailed traffic data, network operators can make informed decisions to optimize the network, such as rerouting traffic to avoid congestion.
- **Troubleshooting**: In case of network issues, these counters provide the necessary data to quickly identify and resolve problems.
- **Policy Validation**: Ensuring that the implemented SRv6 policies are functioning as intended and making adjustments as needed based on the counter data.

# SR-TE Policy Path Protection

*Table 29: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR-TE Policy Path Protection | Release 7.4.2 | You can now configure pre-programmed SR-TE policy Working and Protect candidate paths, and provide fast failure detection through SR Policy Liveness Monitoring probes. If there is a liveness failure on the Working candidate path, the headend triggers a switchover to the Protect candidate path.<br><br>With this release, you can operate IP-centric (with ECMP and TI-LFA) and TDM-centric (with circuits and path protection) services over a common SR network. This eliminates the need for multiple parallel networks and reduces capital expenditures (CapEx) and operating expenditures (OpEx).<br><br>For this feature, the following commands/keywords are added:<br>• **policy path-protection**<br>• **policy candidate-paths preference lock duration**<br>• **backup** keyword is added to the **performance-measurement liveness-detection** command. |

To provide SR policy path protection, headend router and liveness monitoring functions are introduced. The functions are explained with the *1:1* (*one-to-one*) *path protection with SR policy liveness monitoring* use case for TDM-centric networks. Pointers:



**Note** Path protection and local TI-LFA FRR are mutually exclusive functions.

- An SR-TE policy is enabled on the headend router. The headend router 1 sends traffic to endpoint router 7. The Working candidate path **Blue** spans routers 1-2-3-7, and the Protect candidate path **Green** spans routers 1-4-5-6-7.
- The headend Router maintains an independent liveness session on each candidate path using loopback measurement mode. After verifying liveness, it pre-programs Working and Protect paths in forwarding.
- The paths are manually configured in explicit segment lists using MPLS labels to ensure that unprotected adjacency SIDs are utilized.
- The headend router sends traffic over the Working candidate path, and detects any liveness failure. When there is a failure, it sends direct switchover notifications to the FIB, and triggers a switchover to the protected path.
- In 1:1 (***one-to-one***) path protection, when the Working candidate path fails, the Protect candidate path sends traffic.

**Note** SR-TE policy path protection and SR-TE path invalidation drop inter-working is not supported.

### Liveness Monitoring

- SR PM Liveness probes are performed over Working and Protect candidate paths.
- TWAMP Light (RFC 5357) is used for performance measurement and liveness monitoring.
- Separate PM liveness monitoring sessions are created for working and protect candidate-paths.
- Independent PM sessions are created at both endpoints of the SR Policy.
- Loopback measurement-mode (timestamps t1/t4) is used for liveness monitoring. Probe packets are not punted on the responder node. Round-trip delay is computed as (t4 – t1).
- From headend router 1, PM probe query packets are sent with forward and reverse (7->3->2->1) direction paths of the SR Policy's candidate-path in the header of the probe packet. Similarly, PM probe query packets are sent along the Protect path.
- For liveness monitoring:
  - Liveness is declared UP as soon as one probe packet is received back on all segment-lists of the candidate-path.
  - Liveness failure is detected when last N (user-configured value) consecutive probe packets are lost on any segment-list.
  - Fault in the forward and reverse direction of the segment-list (co-routed path) triggers liveness failure notification to SRTE and FIB. FIB triggers protection switchover upon PM notification (running on high priority thread).

### Configuration

- In this example, an SR-TE policy **foo** is created on the headend router and path-protection is enabled for the policy.

```
RP/0/RSP0/CPU0:ios# configure
RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng policy foo
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# color 10 end-point ipv4 192.168.0.3
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# path-protection
RP/0/RSP0/CPU0:ios(config-sr-te-path-pref-protection)#exit
```

- Under **candidate-paths**, the Protect and Working paths are specified through explicit segment lists.
- The Protect path's preference is 50, and it is lower than the Working path preference of 100. The forward (1->4->5->6->7) and reverse (7->6->5->4->1) Protect paths, and the forward (1->2->3->7) and reverse (7->3->2->1) Working paths are enabled as explicit segment lists.
- When the Working path is invalid, the Protect path becomes active. After the Working path has recovered, the Protect path remains active until the default lock duration (of 300 seconds) expires. You can configure a different lock duration using the **lock duration** command.

  The duration range is 0 (disabled) to 3000 seconds. If the lock duration is 0 (disabled), then the Working path becomes active as soon as it recovers. If the duration is not specified, the Protect path remains active.

```
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# candidate-paths
RP/0/RSP0/CPU0:ios(config-sr-te-policy-path)#preference 50
RP/0/RSP0/CPU0:ios(config-sr-te-policy-path-pref)#lock duration 30
RP/0/RSP0/CPU0:ios(config-sr-te-policy-path-pref)# explicit segment-list sl-protect-fwd
```

Type **Exit** three times to go to the SR-TE policy configuration mode.

```
RP/0/RSP0/CPU0:ios(config-sr-te-policy)#candidate-paths
RP/0/RSP0/CPU0:ios(config-sr-te-policy-path)#preference 100
RP/0/RSP0/CPU0:ios(config-sr-te-policy-path-pref)# explicit segment-list sl-working-fwd
RP/0/RSP0/CPU0:ios(config-sr-te-pp-info)# commit
```

### Working and Protect Segment Lists Configuration

Configure explicit segment lists for the candidate paths.

**Note** Segment lists must use only unprotected (dynamic or manual) Adjacency SID and BSIDs (as non-first-SID).

```
RP/0/RSP0/CPU0:ios# configure
RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng
RP/0/RSP0/CPU0:ios(config-sr-te)# segment-list sl-working-fwd
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 1 mpls label 24000
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 2 mpls label 24004
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# exit
RP/0/RSP0/CPU0:ios(config-sr-te)# segment-list sl-working-bck
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 1 mpls label 24002
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 2 mpls label 24006
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# exit
RP/0/RSP0/CPU0:ios(config-sr-te)# segment-list sl-protect-fwd
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 1 mpls label 24000
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 2 mpls label 30201
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# exit
RP/0/RSP0/CPU0:ios(config-sr-te)# segment-list sl-protect-bck
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 1 mpls label 24002
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# index 2 mpls label 30201
RP/0/RSP0/CPU0:ios(config-sr-te-sl)# commit
```

### Performance Measurement Configuration For SR-TE Policy

- Enable SR-TE policy specific performance measurement configurations.
- Create a liveness profile for the Working and Protect paths.

```
RP/0/RSP0/CPU0:ios# configure
RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng policy foo
RP/0/RSP0/CPU0:ios(config-sr-te-policy)#performance-measurement
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)#liveness-detection
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# liveness-profile backup name
profile-PROTECT
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# liveness-profile name profile-WORKING
```

- The default Invalidation action is Down and it triggers path protection switching. The other action is None, which is enabled here.

```
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)#invalidation-action none
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)#commit
```

### Performance Measurement Global Profile Configuration

- Create a Working candidate path liveness profile.

```
RP/0/RSP0/CPU0:ios(config)#performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)#liveness-profile sr-policy name profile-WORKING
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# probe
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# measurement-mode loopback
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tx-interval 30000
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# commit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# liveness-detection multiplier 4
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# commit
```

Type **Exit** to access the Performance Measurement config mode.

- Create a Protect candidate path liveness profile.

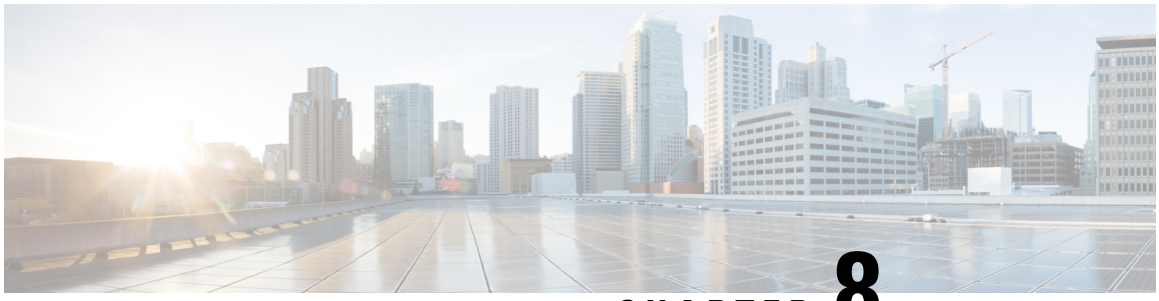
```
RP/0/RSP0/CPU0:ios(config-perf-meas)# liveness-profile sr-policy name profile-PROTECT
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# probe
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# measurement-mode loopback
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tx-interval 100000
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# commit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# liveness-detection multiplier 3
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# commit
```

### Verification

Use the **show segment-routing traffic-eng policy candidate-path** command to display Working and Protect candidate-path details.

```
RP/0/RSP0/CPU0:ios# show segment-routing traffic-eng policy candidate-path name foo

SR-TE policy database
---------------------

Color: 10, End-point: 192.168.0.3s
  Name: srte_c_10_ep_192.168.0.3
  Status:
    Admin: up  Operational: Up for 00:11:55 (since Dec 15 07:02:08.709)
```

```
Candidate-paths:
  Preference: 100 (configuration) (active)
    Name: foo
    Requested BSID: dynamic
      Protection Type: protected-preferred
      Maximum SID Depth: 10
    Explicit: segment-list sl-working-fwd (active)
      Weight: 1, Metric Type: TE
        24000
        24004
    Protection Information:
      Role: WORKING
      Path Lock: Timed
      Lock Duration: 300(s)
  Preference: 50 (configuration) (active)
    Name: foo
    Requested BSID: dynamic
      Protection Type: protected-preferred
      Maximum SID Depth: 10
    Explicit: segment-list sl-protect-fwd (active)
      Weight: 1, Metric Type: TE
        24000
        30201
    Protection Information:
      Role: PROTECT
      Path Lock: Timed
      Lock Duration: 30(s)
..
```



CHAPTER 8

# Segment Routing Tree Segment Identifier

Tree Segment Identifier (Tree-SID) is an SDN controller-based approach to build label switched multicast (LSM) Trees for efficient delivery of multicast traffic in an SR domain and without the need for multicast protocol running in the network. With Tree SID, trees are centrally computed and controlled by a path computation element (SR-PCE).

A Replication segment (as specified in IETF draft "SR Replication segment for Multi-point Service Delivery") is a type of segment which allows a node (Replication node) to replicate packets to a set of other nodes (Downstream nodes) in a Segment Routing Domain.

A Replication segment includes the following:

- Replication SID: The Segment Identifier of a Replication segment. This is an SR-MPLS label (Tree SID label).
- Downstream nodes: Set of nodes in Segment Routing domain to which a packet is replicated by the Replication segment.

A Point-to-Multipoint (P2MP) tree is formed by stitching Replication segments on the Root node, intermediate Replication nodes, and Leaf nodes. This is referred to as an SR P2MP Policy (as specified in IETF draft "Segment Routing Point-to-Multipoint Policy").

An SR P2MP policy works on existing MPLS data-plane and supports TE capabilities and single/multi routing domains. At each node of the tree, the forwarding state is represented by the same Replication segment (using a global Tree-SID specified from the SRLB range of labels).

An SR P2MP policy request contains the following:

- Policy name
- SID for the P2MP Tree (Tree-SID)
- Address of the root node
- Addresses of the leaf nodes
- Optimization objectives (TE, IGP, delay metric)
- Constraints (affinity)

The SR-PCE is responsible for the following:

1. Learning the network topology - *to be added*

2. Learning the Root and Leaves of a Tree - *describe dynamic and static Tree SIDs (16-17) - Tree SID Policy Types and Behaviors*
3. Computing the Tree
4. Allocating MPLS label for the Tree
5. Signaling Tree forwarding state to the routers
6. Re-optimizing Tree

### Tree SID Policy Types and Behaviors

- Static P2MP Policies—can be configured in the following ways:
  - Tree SID parameters provided via Cisco Crosswork Optimization Engine (COE) UI
    - COE passes the policy configuration to the SR-PCE via REST API (no Tree-SID CLI at PCE). This method allows for SR-PCE High Availability (HA).

**Note** Refer to the *Traffic Engineering in Crosswork Optimization Engine* chapter in the Cisco Crosswork Optimization Engine documentation.

  - Tree SID parameters configured via Tree-SID CLI at the SR-PCE

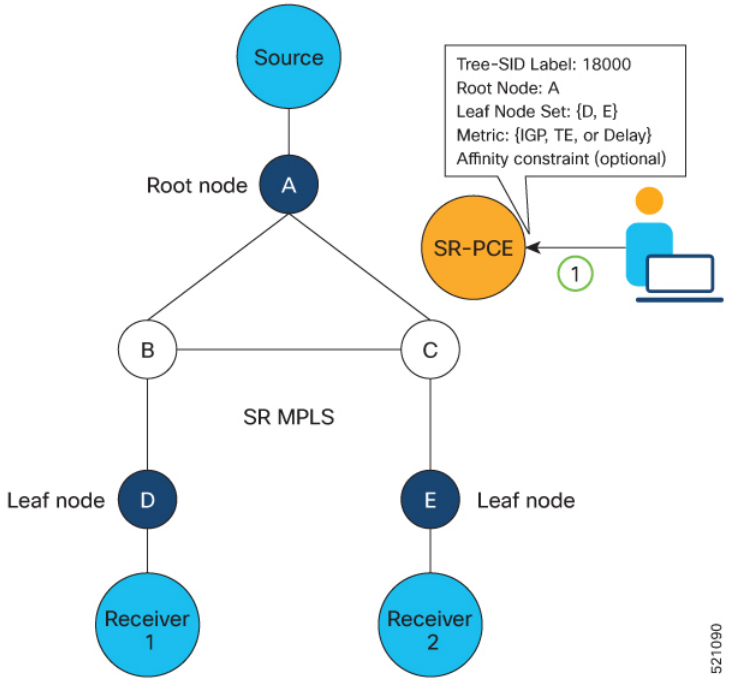
**Caution** With this method, SR-PCE HA is not supported. For this reason, this configuration method is not recommended.

- Dynamic P2MP Policies—can be configured in the following ways:
  - A BGP mVPN is configured in the network (PE nodes) – service configuration via CLI or Cisco NSO
    - As a result, BGP control plane is used for PE auto-discovery and customer multicast signaling.
  - Tree SID parameters are provided by mVPN PEs via PCEP to the PCE. This method allows for SR-PCE High Availability (HA).

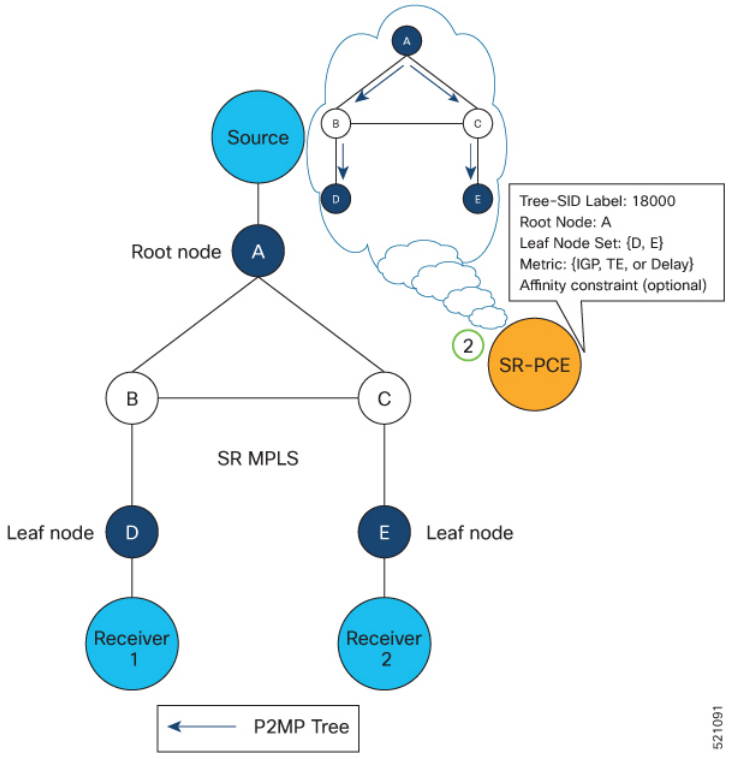
### Tree SID Workflow Overview

This sections shows a basic workflow using a static Tree SID policy:

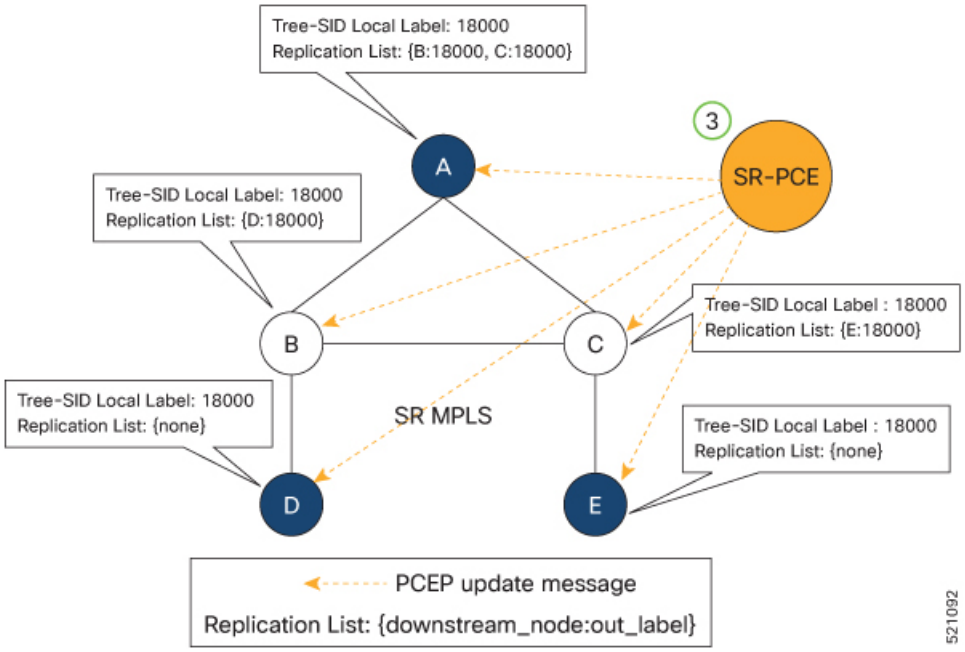
1. User creates a static Tree-SID policy, either via Crosswork Optimization Engine (preferred), or via CLI at the SR-PCE (not recommended).



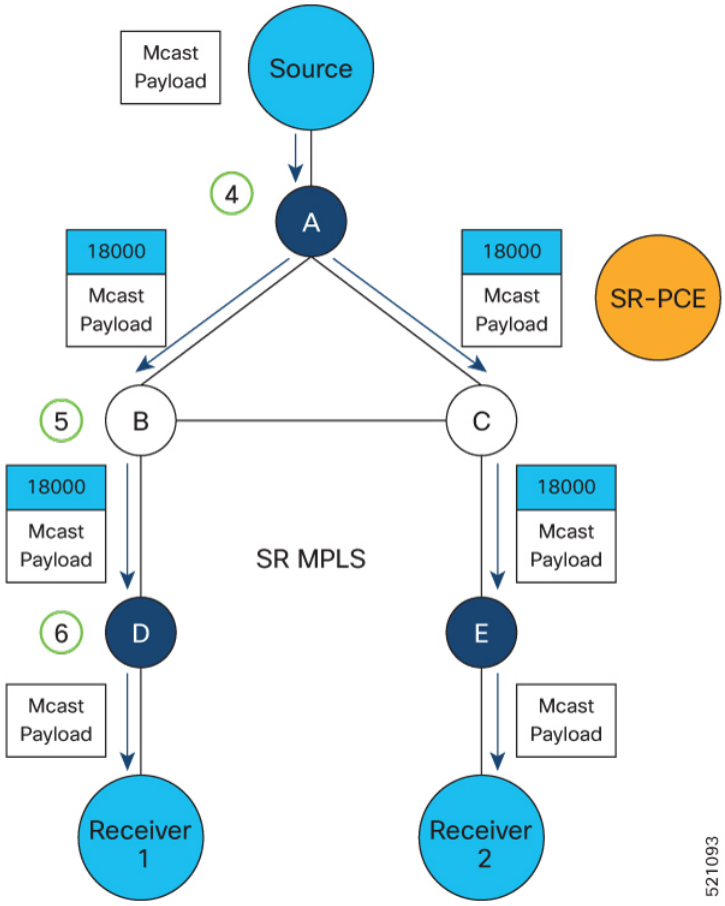

2. SR-PCE computes the P2MP Tree.




3. SR-PCE instantiates the Tree-SID state at each node in the tree.




4. The Root node encapsulates the multicast traffic, replicates it, and forwards it to the Transit nodes.
5. The Transit nodes replicate the multicast traffic and forward it to the Leaf nodes.
6. The Leaf nodes decapsulate the multicast traffic and forward it to the multicast receivers.

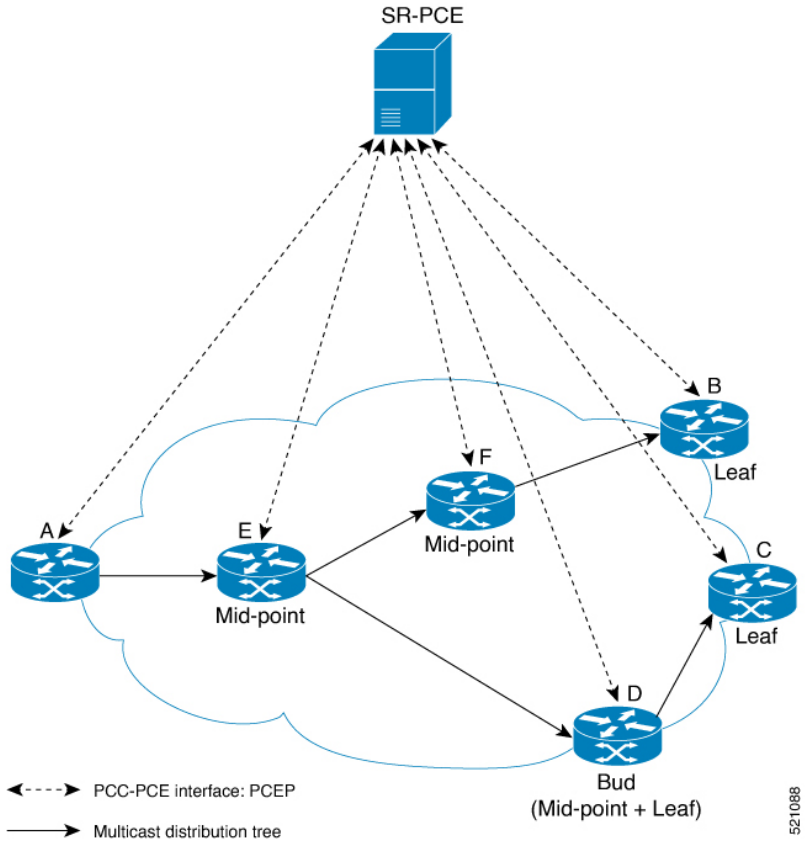





# Bud Node Support

In a multicast distribution tree, a Bud node is a node that acts as a leaf (egress) node as well as a mid-point (transit) node toward the downstream sub-tree.

In the below multicast distribution tree topology with Root node {A} and Leaf nodes set {B, C, D}, node D is a Bud node. Similarly, if node E is later added to the Leaf set, it would also become a Bud node.




The tree computation algorithm on SR-PCE has been enhanced to detect a Bud node based on knowledge of the Leaf set, and to handle Leaf/Transit node transitions to Bud node. The role of the Bud node is also explicitly signaled in PCEP.

# Configure Static Segment Routing Tree-SID via CLI at SR-PCE

**Caution** With this configuration method, SR-PCE HA is not supported. For this reason, this configuration method is not recommended.

To configure static Segment Routing Tree-SID for Point-to-Multipoint (P2MP) SR policies, complete the following configurations:

1. Configure Path Computation Element Protocol (PCEP) Path Computation Client (PCC) on all nodes involved in the Tree-SID path (root, mid-point, leaf)
2. Configure Affinity Maps on the SR-PCE
3. Configure P2MP SR Policy on SR-PCE
4. Configure Multicast on the Root and Leaf Nodes

### Configure PCEP PCC on All Nodes in Tree-SID Path

Configure all nodes involved in the Tree-SID path (root, mid-point, leaf) as PCEP PCC. For detailed PCEP PCC configuration information, see Configure the Head-End Router as PCEP PCC, on page 216.

### Configure Affinity Maps on the SR-PCE

Use the **affinity bit-map** *COLOR bit-position* command in PCE SR-TE sub-mode to define affinity maps. The bit-position range is from 0 to 255.

```
Router# configure
Router(config)# pce
Router(config-pce)# segment-routing traffic-eng
Router(config-pce-sr-te)# affinity bit-map RED 23
Router(config-pce-sr-te)# affinity bit-map BLUE 24
Router(config-pce-sr-te)# affinity bit-map CROSS 25
Router(config-pce-sr-te)#
```

### Configure P2MP SR Policy on SR-PCE

Configure the end-point name and addresses, Tree-SID label, and constraints for the P2MP policy.

Use the **endpoint-set** *NAME* command in SR-PCE P2MP sub-mode to enter the name of the end-point set and to define the set of end-point addresses.

```
Router(config-pce-sr-te)# p2mp
Router(config-pce-sr-te-p2mp)# endpoint-set BAR
Router(config-pce-p2mp-ep-set)# ipv4 10.1.1.2
Router(config-pce-p2mp-ep-set)# ipv4 10.1.1.3
Router(config-pce-p2mp-ep-set)# ipv4 10.1.1.4
Router(config-pce-p2mp-ep-set)# exit
Router(config-pce-sr-te-p2mp)#
```

Use the **policy** *policy* command to configure the P2MP policy name and enter P2MP Policy sub-mode. Configure the source address, endpoint-set color, Tree-SID label, affinity constraints, and metric type.

```
Router(config-pce-sr-te-p2mp)# policy FOO
Router(config-pce-p2mp-policy)# source ipv4 10.1.1.6
Router(config-pce-p2mp-policy)# color 10 endpoint-set BAR
Router(config-pce-p2mp-policy)# treesid mpls 15200
Router(config-pce-p2mp-policy)# candidate-paths
Router(config-pce-p2mp-policy-path)# constraints
Router(config-pce-p2mp-path-const)# affinity
Router(config-pce-p2mp-path-affinity)# exclude BLUE
Router(config-pce-p2mp-path-affinity)# exit
Router(config-pce-p2mp-path-const)# exit
Router(config-pce-p2mp-policy-path)# preference 100
Router(config-pce-p2mp-policy-path-preference)# dynamic
Router(config-pce-p2mp-path-info)# metric type te
Router(config-pce-p2mp-path-info)# root
Router(config)#
```

### Configure Multicast on the Root and Leaf Nodes

On the root node of the SR P2MP segment, use the **router pim** command to enter Protocol Independent Multicast (PIM) configuration mode to statically steer multicast flows into an SR P2MP policy.

**Note** Enter this configuration only on an SR P2MP segment. Multicast traffic cannot be steered into a P2P policy.

```
Router(config)# router pim
Router(config-pim)# vrf name
Router(config-pim-name)# address-family ipv4
Router(config-pim-name-ipv4)# sr-p2mp-policy FOO
Router(config-pim-name-ipv4-srp2mp)# static-group 235.1.1.5 10.1.1.6
Router(config-pim-name-ipv4-srp2mp)# root
Router(config)#
```

On the root and leaf nodes of the SR P2MP tree, use the **mdt static segment-routing** command to configure the multicast distribution tree (MDT) core as Tree-SID from the multicast VRF configuration submode.

```
Router(config)# multicast-routing
Router(config-mcast)# vrf TEST
Router(config-mcast-TEST)# address-family ipv4
Router(config-mcast-TEST-ipv4)# mdt static segment-routing
```

On the leaf nodes of an SR P2MP segment, use the **static sr-policy** *p2mp-policy* command to configure the static SR P2MP Policy from the multicast VRF configuration submode to statically decapsulate multicast flows.

```
Router(config)# multicast-routing
Router(config-mcast)# vrf TEST
Router(config-mcast-TEST)# address-family ipv4
Router(config-mcast-TEST-ipv4)# static sr-policy FOO
```

# Running Config

The following example shows how to configure the end point addresses and P2MP SR policy with affinity constraints on SR-PCE.

```
pce
 segment-routing
  traffic-eng
   affinity bit-map
    RED 23
    BLUE 24
    CROSS 25
   !
   p2mp
    endpoint-set BAR
     ipv4 10.1.1.2
     ipv4 10.1.1.3
     ipv4 10.1.1.4
    !
    policy FOO
     source ipv4 10.1.1.6
     color 10 endpoint-set BAR
     treesid mpls 15200
     candidate-paths
      preference 100
       dynamic
        metric
         type te
```

```
        !
       !
      !
      constraints
       affinity
        exclude
         BLUE
        !
       !
      !
     !
    !
   !
  !
 !
!
```

The following example shows how to statically decapsulate multicast flows on the leaf nodes.

```
multicast-routing
 vrf TEST
  address-family ipv4
   static sr-policy FOO
  !
 !
!
```

The following example shows to configure the multicast distribution tree (MDT) core as Tree-SID on the root and leaf nodes.

```
multicast-routing
 vrf TEST
  address-family ipv4
   mdt static segment-routing
  !
 !
!
```

The following example shows how to steer traffic to the SR P2MP policy on the root node.

```
router pim
 vrf TEST
  address-family ipv4
   sr-p2mp-policy FOO
    static-group 232.1.1.5 10.1.1.6
   !
  !
 !
!
```

# Multicast VPN: Dynamic Tree-SID MVPN (with TI-LFA)

*Table 30: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Multicast VPN: Dynamic Tree-SID MVPN (with TI-LFA) | Release 7.3.1 | With this feature, you can use SR and MVPN for optimally transporting IP VPN multicast traffic over the SP network, using SR-PCE as a controller.<br>With SR's minimal source router configuration requirement, its ability to implement policies with specific optimization objectives and constraints, protect against network failures using TI-LFA FRR mechanism, and use SR-PCE to dynamically generate optimal multicast trees (including when topology changes occur in the multicast tree), the SR-enabled SP network can transport IP multicast traffic efficiently. |

### Prerequisites for Multicast VPN: Tree-SID MVPN With TI-LFA

- The underlay OSPF/IS-IS network is configured, and OSPF/IS-IS adjacency is formed between routers, across the network.
- BGP is configured for the network, and BGP adjacency is formed between routers. BGP MVPN configuration information is provided in this feature document.
- To understand the benefits, know-how, and configuration of SR and SR-TE policies, see About Segment Routing and Configure SR-TE Policies.

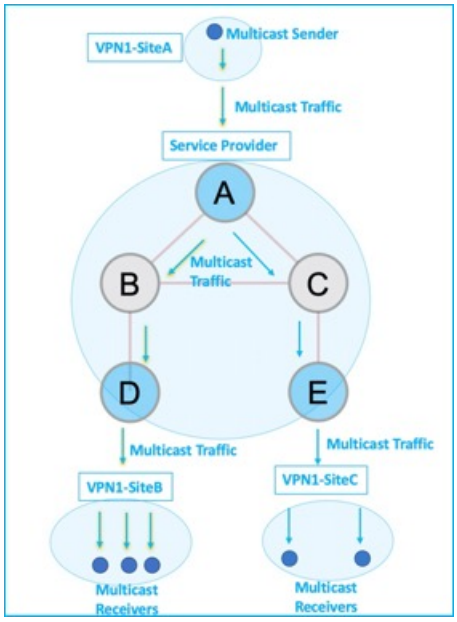
### Information About Multicast VPN: Tree-SID MVPN With TI-LFA

Typically, a customer's IP VPN is spread across VPN sites. IP VPN customer traffic is sent from one site to another over a VPN Service Provider (SP) network.

When IP multicast traffic within a (BGP/MPLS) IP VPN is transported over an SP network (say, from **VPN1-Site-A** to **VPN1-Site-B**, as shown in the image), the SP network requires protocols and procedures to optimally transport multicast traffic from a multicast sender in Site-A to multicast receivers in Site-B.

This use case explains how to enable SR multicast for an SP network, and efficiently transport IP VPN multicast traffic (sent from **VPN1-Site-A** and) received at PE router A, through to PE routers D and E, towards receivers in sites **VPN1-Site-B** and **VPN1-Site-C**.

*Figure 16: IP VPN Multicast Traffic Flow Over An SP Network*




To enable the *Multicast VPN: Tree-SID MVPN With TI-LFA* feature, the following protocols and software applications are used.

**OSPF/IS-IS** - The underlay network is created with OSPF/IS-IS routing protocol, and reachability is established across the network. See *Configure Segment Routing for IS-IS Protocol*or *Configure Segment Routing for OSPF Protocol* chapter for details.

**BGP Multicast VPN (MVPN)** – The PE routers (A, D, and E) are IP VPN end-points for IP multicast traffic arriving at the SP network (at PE router A) and exiting the SP network (at PE routers D and E). So, BGP MVPN is enabled on the PE routers. NSO is used to configure BGP MVPN on the PE routers.

**BGP Auto-Discovery (AD)** - To enable distributed VPN end-point discovery and C-multicast flow mapping and signalling, BGP AD function is configured on the PE routers. A BGP Auto-Discovery route contains multicast router (loopback IP address) and tree identity (segment ID) information. It carries the information in the Provider Multicast Service Interface (PMSI) Tunnel Attribute (PTA).

C-multicast states are signaled using BGP.

**SR** - To transport IP multicast traffic between the VPN end-points (PE routers A, D, and E), Provider (or P-) tunnels are used. In a P-tunnel, the PE devices are the tunnel end-points. P-tunnels can be generated using different technologies (RSVP-TE, P2MP LSPs, PIM trees, mLDP P2MP LSPs, and mLDP MP2MP LSPs). In this use case, Segment Routing (SR) is used for its benefits that were noted earlier.

With SR and SR-PCE, a Tree-SID Point-to-Multipoint (P2MP) segment is used to create P-Tunnels for MVPN. You can specify SR policy optimization objectives (such as *metrics*) and constraints (such as *affinity*) in an SR policy and send it to the SR-PCE controller, so that it can dynamically create SR multicast trees for traffic flow.

**SR-PCE** - This is a controller which, based on the provided SR policy information, computes optimal paths for a multicast tree, and deploys the tree forwarding state on the multicast routers. When a topology change occurs, SR-PCE automatically computes a new, optimal multicast tree, and deploys the new tree forwarding state on the multicast routers.

**TI-LFA** - In SR-TE, Topology-Independent Loop-Free Alternate (TI-LFA) fast reroute (FRR) function is used to reduce link and node failure reaction time. When the primary next-hop (router link) fails, a pre-computed alternate next hop is used to send traffic. TI-LFA FRR is used when transporting IP VPN multicast traffic.

### Overview of Multicast VPN: Tree-SID MVPN With TI-LFA

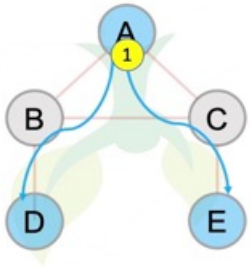
The following sections provide an overview of Tree-SID MVPN and TI-LFA. The topology remains the same, with PE routers A, D, and E acting as VPN end-points for carrying IP VPN multicast traffic.

#### Tree-SID MVPN Overview

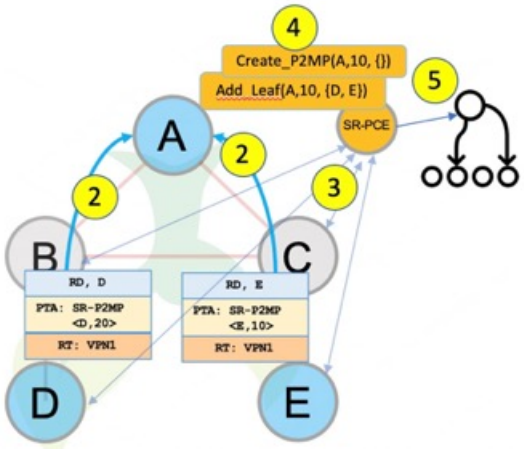
1. For SR, A is designated as the SR head-end router, and D and E are designated as the SR end-points.

   For multicast traffic, A is the root of the SR multicast tree, and D and E are leaf routers of the tree. B and C are the other multicast routers. The objective is to send the IP multicast traffic arriving at A to D and E, as needed

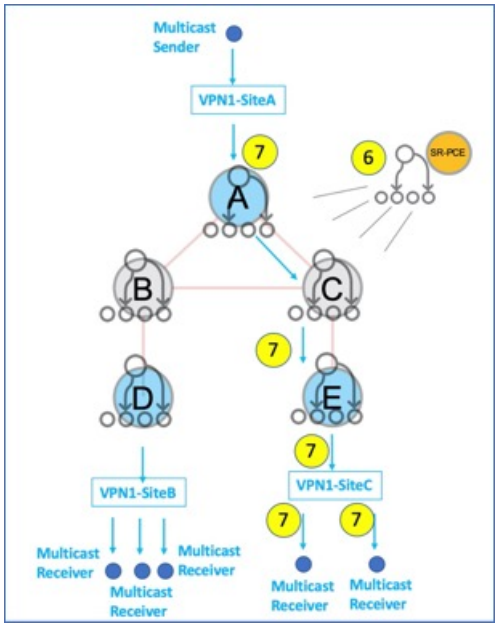
   ***Figure 17: Multicast Tree***




2. A discovers leaf routers' information through BGP MVPN.
3. Path Computation Element Protocol (PCEP) is used for the SR multicast policy communication between A and the SR-PCE server, and communication between PE routers and the SR-PCE server.
4. When the head-end router SR policy is created on A, and PCEP configurations are enabled on the SR-PCE server and all multicast routers, SR-PCE receives the SR policy and leaf router identity information from A.
5. Based on the policy information it receives, including TE objectives and constraints, SR-PCE builds multicast distribution trees in the underlay for efficient VPN traffic delivery.



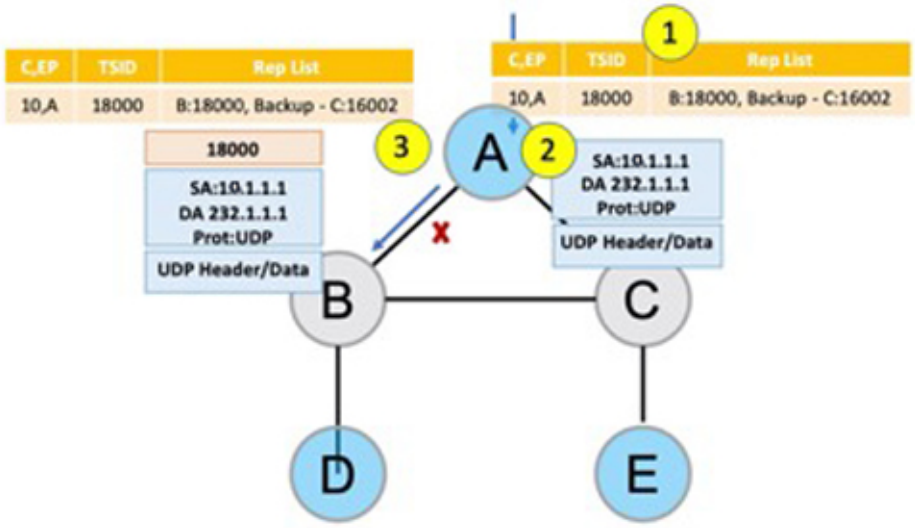
6. SR-PCE assigns an SID for the SR multicast tree policy, and deploys the multicast tree forwarding state on the multicast routers.
7. When IP multicast traffic is sent from VPN1-SiteA to PE router A, it steers it into the SR policy, and sends it towards D and E, which forward it to multicast traffic receivers in the sites VPN1-SiteB and VPN1-SiteC.
8. When a leaf/multicast router is added or removed, PE router A updates the SR multicast policy and sends it to SR-PCE. SR-PCE computes new multicast routes, and deploys the multicast tree forwarding state information on the multicast routers.



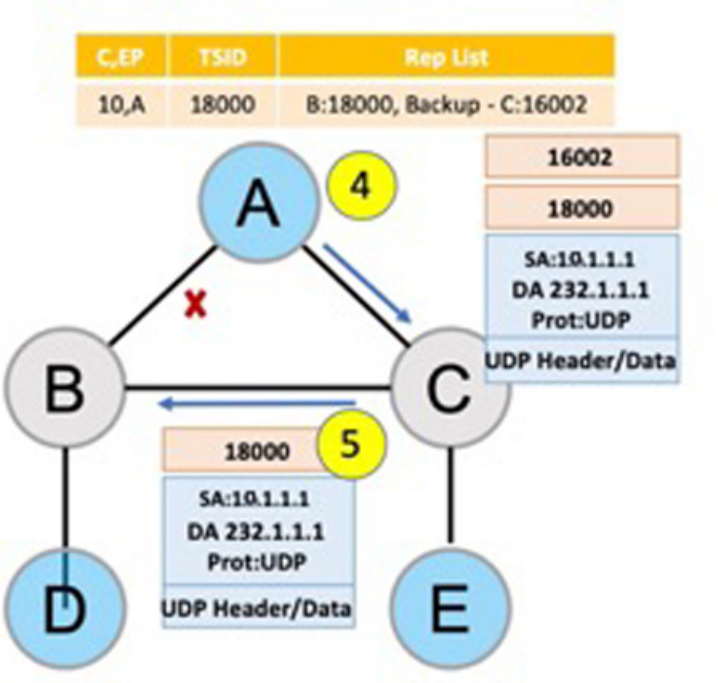

### TI-LFA FRR Overview

High-level TI-LFA FRR function is depicted in these steps:

1. Tree-SID FRR state information.
   - The link from A to B is protected.
   - SID 16002 is the node SID of B.
   - A programs a backup path to B, through C.
2. IP multicast traffic arrives at A which steers the flow onto the tree.
3. A encapsulates and replicates to B, but the link to B is down.



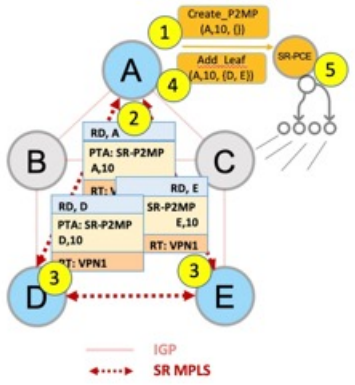

4. A sends the traffic on the backup path, to C.
5. C sends the traffic to B where normal traffic processing resumes.




### SR Multicast Tree Types

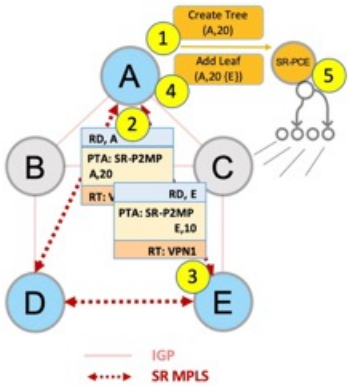
This is an overview of the types of SR multicast trees you can configure, depending on your requirement. You can create a full mesh, on-demand, or optimal multicast tree for IP VPN multicast flow in the SP network.

*Figure 18: Full Mesh Multicast Tree*




1. A assigns Tree-ID 10 and invokes a Create an SR multicast tree request by sending the multicast router and tree ID information (A, 10) towards SR-PCE.
2. A announces BGP AD Inclusive PMSI (I-PMSI) route with the PTA (A, 10). Inclusive PMSI - Traffic that is multicast by a PE router on an I-PMSI is received by all other PEs in the MVPN. I-PMSIs are generated by Inclusive P-tunnels .
3. A discovers VPN endpoints D and E from their BGP AD Type I-PMSI route messages.
4. A invokes an Add SR multicast leaf router request (for D and E) to SR-PCE.
5. SR-PCE computes and generates the multicast tree forwarding state information on all the routers that are part of the tree.

*Figure 19: On-Demand SR Multicast Tree*

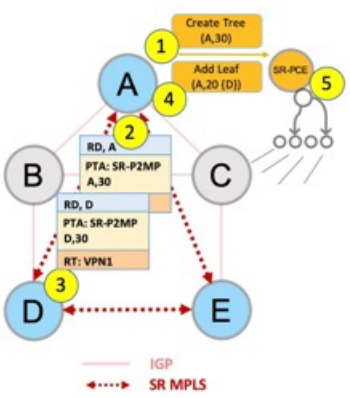



1. A assigns Tree-ID 20 and invokes a Create an SR multicast tree request by sending the multicast router and tree ID information (A, 20) towards SR-PCE.
2. A announces BGP AD Selective PMSI (or S-PMSI) route with PTA (A, 20). A sets the leaf-info-required to discover endpoint interest set.

   *Selective PMSI* - Traffic multicast by a PE on an S-PMSI is received by some PEs in the MVPN. S-PMSIs are generated by Selective P-tunnels.

3. E has a receiver behind it, and announces a BGP-AD leaf route towards A. A discovers service endpoint E for the on-demand tree.
4. A invokes an Add SR multicast leaf router request (for E) to SR-PCE.
5. SR-PCE computes and generates the multicast tree information for all the routers that are part of the tree.

***Figure 20: Optimal Multicast Tree***




1. A decides to optimize a flow and assigns Tree-ID 30 and invokes a Create an SR multicast tree request by sending the multicast router and tree ID information (A, 30) towards SR-PCE.
2. A announces BGP AD I-PMSI route with PTA (A,30). A sets the leaf-info-required to discover endpoint interest set.
3. D has a receiver behind it, and announces a BGP-AD leaf route towards A. A discovers service endpoint D for optimized flow.
4. A invokes an Add SR multicast leaf router request (for D) to SR-PCE.
5. SR-PCE computes and generates the multicast tree information for all the routers that are part of the tree.

### Configurations

**Head End Router Configuration (Router A)** - The following configuration is specific to the head end router.

**Configure TE Constraints and Optimization Parameters**

```
Router# configure terminal
Router(config)# segment-routing traffic-engineering
```

An affinity bit-map is created so that it can be applied to a link or interface.

```
Router(config-sr-te)# affinity-map name 10 bit-position 24
Router(config-sr-te)# commit
```

An affinity (or relationship) is created between the SR policy path and the link color so that SR-TE computes a path that includes or excludes links, as specified. The head-end router automatically follows the actions defined in the ODN template (for color 10) upon the arrival of VPN routes with a BGP color extended community that matches color 10.

```
Router(config)# segment-routing traffic-engineering
Router(config-sr-te)# on-demand color 10 dynamic
Router(config-sr-te-color-dyn)# affinity include-all name red
```

```
Router(config-sr-te-color-dyn)# affinity include-any name blue
Router(config-sr-te-color-dyn)# affinity exclude-any name green
Router(config-sr-te-color-dyn)# metric type te
Router(config-sr-te-color-dyn)# commit
```

The SR policy configuration on the head-end router A will be sent to the SR-PCE server, after a connection is established between A and SR-PCE.

### Multicast Router Configuration

### Configure PCEP Client on Multicast Routers

Associate each multicast router as a client of the SR-PCE server. The **pce address ipv4** command specifies the SR-PCE server's IP address.

```
Router# configure terminal
Router(config)# segment-routing traffic-engineering
Router(config-sr-te)# pcc pce address ipv4 3.3.3.3
Router(config-pcc-pce)# commit
```

### SR PCE Server Configuration

### Configure Label Range for Multicast Trees

Configure the label range to be used for transporting IP multicast traffic in SP network.

```
Router(config)# pce segment-routing traffic-eng p2mp label-range min 30000 max 60000
Router(config)# commit
```

### Configure FRR

The following configurations enable FRR for all SR multicast (P2MP) trees, including dynamic and static implementations.

The **lfa** keyword enables LFA FRR on the PCE server.

```
Router(config)# pce segment-routing traffic-eng p2mp fast-reroute lfa
Router(config)# commit
```

Alternatively, you can configure FRR for each individual tree using the following configuration. The **lfa** keyword under a specific multicast policy (**tree1** in this example) enables LFA FRR function for the specified SR multicast P2MP tree.

For dynamic trees, L-flag in LSP Attributes PCEP object controls FRR on a tree.

```
Router(config)# pce
Router(config-pce)# address ipv4 192.168.0.5
Router(config-pce)# segment-routing traffic-eng p2mp policy tree1 fast-reroute lfa
Router(config-pce)# commit
```

You can create FRR node sets using the **frr-node-set from ipv4** *address* and **frr-node-set to ipv4** *address* commands to specify the *from* and *to* paths on a multicast router that requires FRR protection. In this configuration, the PCE server is configured to manage the FRR function for traffic from 192.168.0.3 sent towards 192.168.0.4 and 192.168.0.5.

```
Router(config)# pce
Router(config-pce)# address ipv4 192.168.0.5
Router(config-pce)# segment-routing traffic-eng
Router(config-pce-sr-te)# p2mp
Router(config-pce-sr-te-p2mp)# frr-node-set from ipv4 192.168.0.3
Router(config-pce-sr-te-p2mp)# frr-node-set to ipv4 192.168.0.4
Router(config-pce-sr-te-p2mp)# frr-node-set to ipv4 192.168.0.5
Router(config-pce-sr-te-p2mp)# commit
```

### Disable ECMP load splitting

To disable ECMP load splitting of different trees on the SR-PCE server, configure the **multipath-disable** command.

```
Router(config)# pce segment-routing traffic-eng p2mp multipath-disable
Router(config)# commit
```

### Multicast Routing Configuration On PE Routers

The following MVPN configurations are required for VPN end-points, the 3 PE routers.

### Configure Default MDT SR P2MP MVPN Profile

In this configuration, an MDT profile of the type *default* is created, and the SR multicast policy with color 10 will be used to send IP multicast traffic, as per the constraints and optimizations of the policy, through the multicast tree.

You can also specify the FRR LFA function with the **mdt default segment-routing mpls fast-reroute lfa** command.

```
Router(config)# multicast-routing vrf cust1
Router(config-mcast-cust1)# address-family ipv4
Router(config-mcast-cust1-ipv4)# mdt default segment-routing mpls color 10
Router(config-mcast-cust1-ipv4)# commit
```

### Configure Partitioned MDT SR P2MP MVPN Profile

In this configuration, an MDT profile of the type *partitioned* is created, and the SR multicast policy with color 10 will be used to send IP multicast traffic, as per the constraints and optimizations of the policy, through the multicast tree.

You can also specify the FRR LFA function with the **mdt partitioned segment-routing mpls fast-reroute lfa** command.

```
Router(config)# multicast-routing vrf cust1
Router(config-mcast-cust1)# address-family ipv4
Router(config-mcast-cust1-ipv4)# mdt partitioned segment-routing mpls color 10
Router(config-mcast-cust1-ipv4)# commit
```

The following Data MVPN configuration is required at the Ingress PE (router A) where the multicast flows need to be steered onto the *data* MDT for SR multicast traffic flow.

**Note** - *Data* MDT can be configured for *Default* and *Partitioned* profiles.

### Configure Data MDT for SR P2MP MVPN

In this configuration, an MDT profile of the type *data* is created, and the SR multicast policy with color 10 will be used to send IP multicast traffic, as per the constraints and optimizations of the policy, through the multicast tree.

- You can enable the FRR LFA function with the **mdt data segment-routing mpls fast-reroute lfa** command. This enables LFA FRR for SR multicast trees created for all data MDT profiles.
- As an alternative to the color keyword, you can specify a route policy in the **route-policy** command, and define the route policy separately (as mentioned in the next configuration).
- The **threshold** command specifies the threshold above which a multicast flow is switched onto the data MDT. The **immediate-switch** keyword enables an immediate switch of a multicast flow to the data MDT, without waiting for threshold limit to be crossed.
- The **customer-route-acl** keyword specifies an ACL to enable specific multicast flows to be put on to the data MDT.

- **color** and **fast-reroute lfa** keywords are mutually exclusive with the **route-policy** configuration. The objective is to apply constraints (through **color**) or FRR (through LFA protection) to either all data MDTs, or apply them selectively per data MDT, using the **set on-demand-color** and **set fast-reroute lfa** options in the route policy (configured in the **mdt data** configuration).

```
Router(config)# multicast-routing vrf cust1
Router(config-mcast-cust1)# address-family ipv4
Router(config-mcast-cust1-ipv4)# mdt data segment-routing mpls 2 color 10
Router(config-mcast-cust1-ipv4)# commit
```

**Route Policy Example**

The route policy designates multicast flow-to-SR multicast policy mapping, with different colors.

- With this configuration, IP multicast flows for the 232.0.0.1 multicast group are steered into the SR multicast policy created with the on-demand color 10, while flows for 232.0.0.2 are steered into the policy created with color 20.
- The *data* MDT SR multicast tree created for the 232.0.0.2 multicast group is enabled with FRR LFA protection.
- Route policies can also be used to match other parameters, such as source address.

```
Router(config)# route-policy TSID-DATA
Router(config-rpl)# if destination in (232.0.0.1) then
Router(config-rpl-if)# set on-demand-color 10
Router(config-rpl-if)# pass
Router(config-rpl-if)# elseif destination in (232.0.0.2) then
Router(config-rpl-elseif)# set on-demand-color 20
Router(config-rpl-elseif)# set fast-reroute lfa
Router(config-rpl-elseif)# pass
Router(config-rpl-elseif)# endif
Router(config-rpl)# end-policy
Router(config)# commit
```

**Configure MVPN BGP Auto-Discovery for SR P2MP**

The following configuration is required on all PE routers, and is mandatory for *default* MDT, *partitioned* MDT, and *data* MDT.

Configure the BGP Auto-Discovery function for transporting IP multicast traffic.

```
Router(config)# multicast-routing vrf cust1
Router(config-mcast-cust1)# address-family ipv4
Router(config-mcast-cust1-ipv4)# bgp auto-discovery segment-routing
Router(config-mcast-cust1-ipv4-bgp-ad)# commit
```

### Verification

**View MVPN Context Information** - You can view MVPN VRF context information with these commands.

**View Default MDT Configuration**

This command displays SR multicast tree information, including the MDT details (of *default* type, etc), and customer VRF information (route target, route distinguisher, etc).

```
Router# show mvpn vrf vpn1 context

MVPN context information for VRF vpn1 (0x9541cf0)

RD: 1:10 (Valid, IID 0x1), VPN-ID: 0:0
Import Route-targets : 2
```

```
  RT:192.168.0.4:0, BGP-AD
  RT:192.168.0.4:17, BGP-AD
BGP Auto-Discovery Enabled (I-PMSI added)
SR P2MP Core-tree data:
  MDT Name: TRmdtvpn1, Handle: 0x4150, idb: 0x956fc30
  MTU: 1376, MaxAggr: 255, SW_Int: 30, AN_Int: 60
  RPF-ID: 3, C:0, O:1, D:0, CP:0
  Static Type : - / -
  Def MDT ID: 524289 (0x93993f0), added: 1, HLI: 0x80001, Cfg: 1/0
  Part MDT ID: 0 (0x0), added: 0, HLI: 0x00000, Cfg: 0/0
  Ctrl Trees : 0/0/0, Ctrl ID: 0 (0x0), Ctrl HLI: 0x00000
```

### View Partitioned MDT Configuration

This command displays SR multicast tree information, including the MDT details (of *partitioned* type, etc), and customer VRF information (route target, route distinguisher, etc).

```
Router# show mvpn vrf vpn1 context

MVPN context information for VRF vpn1 (0x9541cf0)

RD: 1:10 (Valid, IID 0x1), VPN-ID: 0:0
Import Route-targets : 2
  RT:192.168.0.4:0, BGP-AD
  RT:192.168.0.4:17, BGP-AD
BGP Auto-Discovery Enabled (I-PMSI added) , MS-PMSI sent
SR P2MP Core-tree data:
  MDT Name: TRmdtvpn1, Handle: 0x4210, idb: 0x956fc30
  MTU: 1376, MaxAggr: 255, SW_Int: 30, AN_Int: 60
  RPF-ID: 1, C:0, O:1, D:0, CP:0
  Static Type : - / -
  Def MDT ID: 0 (0x0), added: 0, HLI: 0x00000, Cfg: 0/0
  Part MDT ID: 524292 (0x9399318), added: 1, HLI: 0x80004, Cfg: 1/0
  Ctrl Trees : 0/0/0, Ctrl ID: 0 (0x0), Ctrl HLI: 0x00000
```

### View Partitioned MDT Ingress PE Configuration

This command displays SR multicast tree information on the PE router that receives the multicast traffic on the SP network. The information includes PE router details, MDT details, Tree-SID details, and the specified customer VRF information.

```
Router# show mvpn vrf vpn1 pe

MVPN Provider Edge Router information

VRF : vpn1

PE Address : 192.168.0.3 (0x9570240)
  RD: 0:0:0 (null), RIB_HLI 0, RPF-ID 13, Remote RPF-ID 0, State: 0, S-PMSI: 2
  PPMP_LABEL: 0, MS_PMSI_HLI: 0x00000, Bidir_PMSI_HLI: 0x00000, MLDP-added: [RD 0, ID 0,
Bidir ID 0, Remote Bidir ID 0], Counts(SHR/SRC/DM/DEF-MD): 0, 0, 0, 0, Bidir: GRE RP Count
 0, MPLS RP Count 0RSVP-TE added: [Leg 0, Ctrl Leg 0, Part tail 0 Def Tail 0, IR added:
[Def Leg 0, Ctrl Leg 0, Part Leg 0, Part tail 0, Part IR Tail Label 0
  Tree-SID Added: [Def/Part Leaf 1, Def Egress 0, Part Egress 0, Ctrl Leaf 0]
  bgp_i_pmsi: 1,0/0 , bgp_ms_pmsi/Leaf-ad: 1/1, bgp_bidir_pmsi: 0, remote_bgp_bidir_pmsi:
0, PMSIs: I 0x9570378, 0x0, MS 0x94e29d0, Bidir Local: 0x0, Remote: 0x0, BSR/Leaf-ad 0x0/0,
 Autorp-disc/Leaf-ad 0x0/0, Autorp-ann/Leaf-ad 0x0/0
 IIDs: I/6: 0x1/0x0, B/R: 0x0/0x0, MS: 0x1, B/A/A: 0x0/0x0/0x0

  Bidir RPF-ID: 14, Remote Bidir RPF-ID: 0
  I-PMSI:   Unknown/None (0x9570378)
  I-PMSI rem:  (0x0)
  MS-PMSI:   Tree-SID [524290, 192.168.0.3] (0x94e29d0)
  Bidir-PMSI:  (0x0)
  Remote Bidir-PMSI:  (0x0)
```

```
  BSR-PMSI:  (0x0)
  A-Disc-PMSI:  (0x0)
  A-Ann-PMSI:  (0x0)
  RIB Dependency List: 0x0
  Bidir RIB Dependency List: 0x0
    Sources: 0, RPs: 0, Bidir RPs: 0
```

### View Partitioned MDT Egress PE Configuration

This command displays SR multicast tree information on the MVPN egress PE router that sends multicast traffic from the SP network towards multicast receivers in the destination sites. The information includes PE router, Tree-SID, MDT, and the specified customer VRF details.

```
Router# show mvpn vrf vpn1 pe

MVPN Provider Edge Router information

PE Address : 192.168.0.4 (0x9fa38f8)
  RD: 1:10 (valid), RIB_HLI 0, RPF-ID 15, Remote RPF-ID 0, State: 1, S-PMSI: 2
  PPMP_LABEL: 0, MS_PMSI_HLI: 0x00000, Bidir_PMSI_HLI: 0x00000, MLDP-added: [RD 0, ID 0,
Bidir ID 0, Remote Bidir ID 0], Counts(SHR/SRC/DM/DEF-MD): 1, 1, 0, 0, Bidir: GRE RP Count
 0, MPLS RP Count 0RSVP-TE added: [Leg 0, Ctrl Leg 0, Part tail 0 Def Tail 0, IR added:
[Def Leg 0, Ctrl Leg 0, Part Leg 0, Part tail 0, Part IR Tail Label 0
  Tree-SID Added: [Def/Part Leaf 0, Def Egress 0, Part Egress 1, Ctrl Leaf 0]
  bgp_i_pmsi: 1,0/0 , bgp_ms_pmsi/Leaf-ad: 1/0, bgp_bidir_pmsi: 0, remote_bgp_bidir_pmsi:
0, PMSIs: I 0x9f77388, 0x0, MS 0x9fa2f98, Bidir Local: 0x0, Remote: 0x0, BSR/Leaf-ad 0x0/0,
 Autorp-disc/Leaf-ad 0x0/0, Autorp-ann/Leaf-ad 0x0/0
 IIDs: I/6: 0x1/0x0, B/R: 0x0/0x0, MS: 0x1, B/A/A: 0x0/0x0/0x0

  Bidir RPF-ID: 16, Remote Bidir RPF-ID: 0
  I-PMSI:   Unknown/None (0x9f77388)
  I-PMSI rem:  (0x0)
  MS-PMSI:   Tree-SID [524292, 192.168.0.4] (0x9fa2f98)
  Bidir-PMSI:  (0x0)
  Remote Bidir-PMSI:  (0x0)
  BSR-PMSI:  (0x0)
  A-Disc-PMSI:  (0x0)
  A-Ann-PMSI:  (0x0)
  RIB Dependency List: 0x9f81370
  Bidir RIB Dependency List: 0x0
    Sources: 1, RPs: 1, Bidir RPs: 0
```

### View Data MDT Information

The commands in this section displays SR multicast tree information for *data* MDTs. The information includes cache, router-local, and remote MDT information.

### View Data MDT Cache Information

```
Router# show pim vrf vpn1 mdt cache
Core Source      Cust (Source, Group)               Core Data        Expires
192.168.0.3      (26.3.233.1, 232.0.0.1)            [tree-id 524292]  never

192.168.0.4      (27.3.233.6, 232.0.0.1)            [tree-id 524290]  never

   Leaf AD:  192.168.0.3
```

### View Local MDTs Information

```
Router# show pim vrf vpn1 mdt sr-p2mp local

Tree                          MDT            Cache DIP Local VRF Routes  On-demand
Identifier                    Source         Count     Entry Using Cache Color
[tree-id 524290 (0x80002)]    192.168.0.4    1     N   Y     1           10
   Tree-SID Leaf:  192.168.0.3
```

### View Remote MDTs Information

```
Router # show pim vrf vpn1 mdt sr-p2mp remote

Tree                            MDT             Cache DIP Local VRF Routes  On-demand
Identifier                      Source          Count     Entry Using Cache Color
[tree-id 524290 (0x80002)]      192.168.0.4     1     N   N     1           0
```

### View MRIB MPLS Forwarding Information

This command displays labels used for transporting IP multicast traffic, on a specified router.

```
Router# show mrib mpls forwarding

LSP information (XTC) :
  LSM-ID: 0x00000, Role: Head, Head LSM-ID: 0x80002
    Incoming Label       : (18000)
    Transported Protocol : <unknown>
    Explicit Null        : None
    IP lookup            : disabled

    Outsegment Info #1 [H/Push, Recursive]:
      OutLabel: 18000, NH: 192.168.0.3, Sel IF: GigabitEthernet0/2/0/0

LSP information (XTC) :
  LSM-ID: 0x00000, Role: Tail, Peek
  RPF-ID: 0x00011, Assoc-TIDs: 0xe0000011/0x0, MDT: TRmdtvpn1
    Incoming Label       : 18001
    Transported Protocol : <unknown>
    Explicit Null        : None
    IP lookup            : enabled

    Outsegment Info #1 [T/Pop]:
      No info.
```

### SR-PCE Show Commands

### View Tree Information On PCE Server

This command displays SR multicast tree information on the SR-PCE server.

```
Router# show pce lsp p2mp

Tree: sr_p2mp_root_192.168.0.1_tree_id_524290
 Label:    18000     Operational: up  Admin: up
 Metric Type: TE
 Transition count: 3
 Uptime: 00:00:03 (since Fri Jan 24 14:57:51 PST 2020)
 Source: 192.168.0.1
 Destinations: 192.168.0.4
 Nodes:
  Node[0]: 192.168.0.2 (rtrM)
   Role: Transit
   Hops:
    Incoming: 18000 CC-ID: 4
    Outgoing: 18000 CC-ID: 4 (17.17.17.4) [rtrR]
  Node[1]: 192.168.0.1 (rtrL1)
   Role: Ingress
   Hops:
    Incoming: 18000 CC-ID: 5
    Outgoing: 18000 CC-ID: 5 (12.12.12.2) [rtrM]
  Node[2]: 192.168.0.4 (rtrR)
   Role: Egress
   Hops:
    Incoming: 18000 CC-ID: 6
```

For dynamic SR multicast trees created for MVPN, the **show** command has filters to view root multicast router and Tree-ID information. When the root router is specified, all multicast trees from that root are displayed. When root and Tree-ID are specified, only the specified tree information is displayed.

```
Router# show pce lsp p2mp root ipv4 10.1.1.1 524289

Tree: sr_p2mp_root_10.1.1.1_tree_id_524289, Root: 10.1.1.1 ID: 524289
 Label:    20000     Operational: up  Admin: up
 PCC: 10.1.1.1

 Local LFA FRR: Disabled
 Metric Type: TE
 Transition count: 11
 Uptime: 00:03:37 (since Mon May 11 12:53:33 PDT 2020)
 Destinations: 10.1.1.3, 10.1.1.4, 10.1.1.5
 Nodes:
  Node[0]: 10.1.1.1 (root1)
   Role: Ingress
   Hops:
    Incoming: 20000 CC-ID: 26
    Outgoing: 20000 CC-ID: 26 (192.168.114.4) [mid-4]
    Outgoing: 20000 CC-ID: 26 (192.168.112.2) [mid-2]
  Node[1]: 10.1.1.4 (mid-4)
   Role: Egress
   Hops:
    Incoming: 20000 CC-ID: 27
  Node[2]: 10.1.1.2 (mid-2)
   Role: Transit
   Hops:
    Incoming: 20000 CC-ID: 28
    Outgoing: 20000 CC-ID: 28 (192.168.123.3) [leaf-3]
    Outgoing: 20000 CC-ID: 28 (192.168.125.5) [leaf-5]
  Node[3]: 10.1.1.3 (leaf-3)
   Role: Egress
   Hops:
    Incoming: 20000 CC-ID: 29
  Node[4]: 10.1.1.5 (leaf-5)
   Role: Egress
   Hops:
    Incoming: 20000 CC-ID: 30
```

The following output shows that LFA FRR is enabled on the hop from rtrR to rtrM. Unlike typical multicast replication where the address displayed is the remote address on the link to a downstream router, the IP address 192.168.0.3 (displayed with an exclamation mark) is the router-ID of the downstream router rtrM. The output also displays the LFA FRR state for the multicast tree.

```
Router# show pce lsp p2mp

Tree: sr_p2mp_root_192.168.0.4_tree_id_524290
 Label:    18000     Operational: up  Admin: up
 LFA FRR: Enabled
 Metric Type: TE
 Transition count: 1
 Uptime: 3d19h (since Thu Feb 13 13:43:40 PST 2020)
 Source: 192.168.0.4
 Destinations: 192.168.0.1, 192.168.0.2
 Nodes:
  Node[0]: 192.168.0.3 (rtrM)
   Role: Transit
   Hops:
    Incoming: 18000 CC-ID: 1
    Outgoing: 18000 CC-ID: 1 (12.12.12.1) [rtrL1]
    Outgoing: 18000 CC-ID: 1 (15.15.15.2) [rtrL2]
  Node[1]: 192.168.0.4 (rtrR)
```

```
   Role: Ingress
   Hops:
    Incoming: 18000 CC-ID: 2
    Outgoing: 18000 CC-ID: 2 (192.168.0.3!) [rtrM]
  Node[2]: 192.168.0.1 (rtrL1)
   Role: Egress
   Hops:
    Incoming: 18000 CC-ID: 3
  Node[3]: 192.168.0.2 (rtrL2)
   Role: Egress
   Hops:
    Incoming: 18000 CC-ID: 4
```

### Multicast Tree Information on Routers

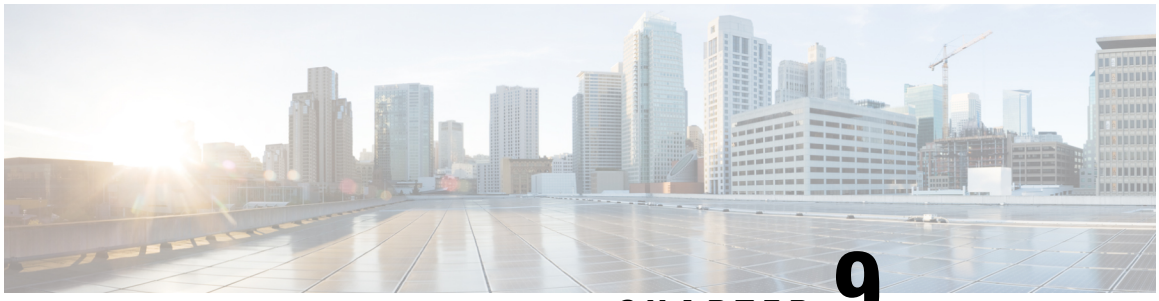
```
Router# show segment-routing traffic-eng p2mp policy

SR-TE P2MP policy database:
---------------------
! - Replications with Fast Re-route

Policy: sr_p2mp_root_192.168.0.1_tree_id_524290  LSM-ID: 0x2
 Role: Leaf
 Replication:
  Incoming label: 18001 CC-ID: 6

Policy: sr_p2mp_root_192.168.0.4_tree_id_524290  LSM-ID: 0x80002 (PCC-initiated)
 Color: 0
 LFA FRR: Disabled
 Role: Root
 Replication:
  Incoming label: 18000 CC-ID: 2
  Interface: None [192.168.0.3!]  Outgoing label: 18000 CC-ID: 2
 Endpoints:
  192.168.0.1, 192.168.0.2
```

For SR multicast policies originated locally on the router (root router of a dynamic MVPN multicast policy) additional policy information is displayed. The information includes color, end points, and whether LFA FRR is requested by the local application. When the SR-PCE server enables LFA FRR on a specific hop, the outgoing information shows the address of the next router with an exclamation mark and None is displayed for the outgoing interface.

For dynamic SR multicast trees created for MVPN, the **show** command has filters for displaying root multicast router and Tree-ID information. When the root router is specified, all multicast trees for that root are displayed. When root and Tree-ID are specified, only the specified tree information is displayed.

```
Router# show segment-routing traffic-eng p2mp policy root ipv4 1.1$

SR-TE P2MP policy database:
---------------------
! - Replications with Fast Re-route, * - Stale dynamic policies/endpoints

Policy: sr_p2mp_root_10.1.1.1_tree_id_524289  LSM-ID: 0x691
 Root: 10.1.1.1, ID: 524289
 Role: Transit
 Replication:
  Incoming label: 20000 CC-ID: 28
  Interface: Bundle-Ether23 [192.168.123.3]  Outgoing label: 20000 CC-ID: 28
  Interface: Bundle-Ether25 [192.168.125.5]  Outgoing label: 20000 CC-ID: 28

Policy: sr_p2mp_root_10.1.1.1_tree_id_524290  LSM-ID: 0x692
 Root: 10.1.1.1, ID: 524290
 Role: Transit
 Replication:
```

```
Incoming label: 19999 CC-ID: 28
Interface: Bundle-Ether23 [192.168.123.3]  Outgoing label: 19999 CC-ID: 28
Interface: Bundle-Ether25 [192.168.125.5]  Outgoing label: 19999 CC-ID: 28
```



CHAPTER 9

# 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.



## 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.

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.

### TCP Authentication Option

TCP Message Digest 5 (MD5) authentication has been used for authenticating PCEP (TCP) sessions by using a clear text or encrypted password. This feature introduces support for TCP Authentication Option (TCP-AO), which replaces the TCP MD5 option.

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

- Protection against replays for long-lived TCP connections
- More details on the security association with TCP connections than TCP MD5
- A larger set of MACs with minimal system and operational changes

TCP-AO is compatible with Master Key Tuple (MKT) configuration. TCP-AO also protects connections when using the same MKT across repeated instances of a connection. TCP-AO protects the connections by using traffic key that are derived from the MKT, and then coordinates changes between the endpoints.

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

# Usage Guidelines and Limitations

To ensure PCEP compatibility, we recommend that the Cisco IOS XR version on the SR-PCE be the same or later than the Cisco IOS XR version on the PCC or head-end.

# Configure SR-PCE

This task explains how to configure SR-PCE.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **pce**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# pce` | Enables PCE and enters PCE configuration mode. |
| **Step 3** | **address ipv4** *address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# address ipv4 192.168.0.1` | Configures a PCE IPv4 address. |
| **Step 4** | **state-sync ipv4** *address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# state-sync ipv4 192.168.0.3` | Configures the remote peer for state synchronization. |
| **Step 5** | **tcp-buffer size** *size*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# tcp-buffer size 1024000` | Configures the transmit and receive TCP buffer size for each PCEP session, in bytes. The default buffer size is 256000. The valid range is from 204800 to 1024000. |
| **Step 6** | **password** {**clear** \| **encrypted**} *password*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# password encrypted pwd1` | Enables TCP MD5 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.<br>**Note** TCP-AO and TCP MD5 are never permitted to be used simultaneously. |
| **Step 7** | **tcp-ao** *key-chain* [**include-tcp-options**] [**accept-ao-mismatch-connection**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# tcp-ao` | Enables TCP Authentication Option (TCP-AO) authentication for all PCEP peers. Any TCP segment coming from the PCC that does not contain a MAC matching the configured key chain will be rejected. |

| | Command or Action | Purpose |
|---|---|---|
| | `pce_tcp_ao include-tcp-options` | • **include-tcp-options**—Includes other TCP options in the header for MAC calculation.<br>• **accept-ao-mismatch-connection**—Accepts connection even if there is a mismatch of AO options between peers.<br>**Note** TCP-AO and TCP MD5 are never permitted to be used simultaneously. |
| **Step 8** | **segment-routing** {**strict-sid-only** \| **te-latency**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# segment-routing strict-sid-only` | Configures the segment routing algorithm to use strict SID or TE latency.<br>**Note** This setting is global and applies to all LSPs that request a path from this controller. |
| **Step 9** | **timers**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# timers` | Enters timer configuration mode. |
| **Step 10** | **keepalive** *time*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce-timers)# keepalive 60` | Configures the timer value for locally generated keep-alive messages. The default time is 30 seconds. |
| **Step 11** | **minimum-peer-keepalive** *time*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce-timers)# minimum-peer-keepalive 30` | Configures the minimum acceptable keep-alive timer that the remote peer may propose in the PCEP OPEN message during session establishment. The default time is 20 seconds. |
| **Step 12** | **reoptimization** *time*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce-timers)# reoptimization 600` | Configures the re-optimization timer. The default timer is 1800 seconds. |
| **Step 13** | **exit**<br>**Example:** | Exits timer configuration mode and returns to PCE configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-pce-timers)# exit` | |

# 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **disjoint-path**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce)# disjoint-path` | Enters disjoint configuration mode. |
| **Step 2** | **group-id** *value* **type** {**link** \| **node** \| **srlg** \| **srlg-node**} [**sub-id** *value*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce-disjoint)# group-id 1 type node sub-id 1` | Configures the disjoint group ID and defines the preferred level of disjointness (the type of resources that should not be shared by the two paths):<br>• **link**—Specifies that links are not shared on the computed paths.<br>• **node**—Specifies that nodes are not shared on the computed paths.<br>• **srlg**—Specifies that links with the same SRLG value are not shared on the computed paths.<br>• **srlg-node**—Specifies that SRLG and nodes are not shared on the computed paths.<br>If a pair of paths that meet the requested disjointness level cannot be found, then the paths will automatically fallback to a lower level:<br>• If the requested disjointness level is SRLG or node, then link-disjoint paths will be computed. |

| | Command or Action | Purpose |
|---|---|---|
| | | • 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**<br>**Example:**<br>`RP/0/RP0/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**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-pce-disjoint)# lsp 1 pcc ipv4 192.168.0.1 lsp-name rtrA_t1 shortest-path`<br>`RP/0/RP0/CPU0:router(config-pce-disjoint)# lsp 2 pcc ipv4 192.168.0.5 lsp-name rtrE_t2` | Adds LSPs to the disjoint group.<br>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 the first LSP specified. |

# PCE-Initiated SR Policies

Use cases based on centralized optimization, such as congestion mitigation solutions, rely on the ability of the PCE to signal and instantiate SR-TE policies in the network. We refer to this as PCE-initiated SR-TE policies.

PCE-initiated SR-TE policies can be triggered via Crossworks Network Controller (recommended approach) or via CLI at the PCE.

For more information on configuring SR-TE policies, see the SR-TE Policy Overview, on page 163.

The 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: PCE-Initiated SR Policy with Explicit SID List

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

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

**Note** When configuring an explicit path using IP addresses of intermediate links, the SR-TE process prefers the protected Adj-SID of the link, if one is available.

3. Create the policy.

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

/* Create the SR-TE segment lists */
Router(config-pce)# segment-routing
Router(config-pce-sr)# traffic-eng
Router(config-pce-sr-te)# segment-list name addr2a
Router(config-pce-sr-te-sl)# index 10 address ipv4 10.1.1.2
Router(config-pce-sr-te-sl)# index 20 address ipv4 10.2.3.2
Router(config-pce-sr-te-sl)# index 30 address ipv4 10.1.1.4
Router(config-pce-sr-te-sl)# exit

/* Create the SR-TE policy */
Router(config-pce-sr-te)# peer ipv4 10.1.1.1
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-policy-path-preference)# explicit segment-list addr2a
Router(config-pce-sr-te-pp-info)# commit
Router(config-pce-sr-te-pp-info)# end
Router(config)#
```

### Running Config

```
pce
 segment-routing
  traffic-eng
   segment-list name addr2a
    index 10 address ipv4 10.1.1.2
    index 20 address ipv4 10.2.3.2
    index 30 address ipv4 10.1.1.4
   !
   peer ipv4 10.1.1.1
    policy P1
     color 2 end-point ipv4 2.2.2.2
     candidate-paths
      preference 50
       explicit segment-list addr2a
      !
     !
```

# SR-PCE Flexible Algorithm Multi-Domain Path Computation

Flexible Algorithm provides a traffic engineered path automatically computed by the IGP to any destination reachable by the IGP. With the SR-PCE Flexible Algorithm Multi-Domain Path Computation feature, SR-PCE can use Flexible Algorithms to compute multi-domain paths. See the Enabling Segment Routing Flexible Algorithm, on page 409 chapter for information about Segment Routing Flexible Algorithm.

The SR-PCE Flexible Algorithm Multi-Domain Path Computation feature incorporates the following functionality:

- BGP-LS has been augmented to allow selected nodes to advertise the Flexible Algorithm definition (FAD) to the SR-PCE
- PCEP has been augmented (vendor-specific object) to allow a PCC to indicate SR policy constraint based on the Flexible Algorithm instance number
- SR-PCE algorithms have been augmented to compute paths based on a Flexible Algorithm constraint

The SR-PCE Flexible Algorithm multi-domain path computation requires the following:

- The same Flexible Algorithm instance ID is used across domains.
- The metric for those Flexible Algorithm instances must be the same across domains.
- The affinity constraints for those Flexible Algorithm instances may be different across domains.
- Multiple Flexible Algorithms can exist in a domain.

For example, considering a multi-domain topology (Domain 1 and Domain 2), the following scenarios meet the requirements listed above:

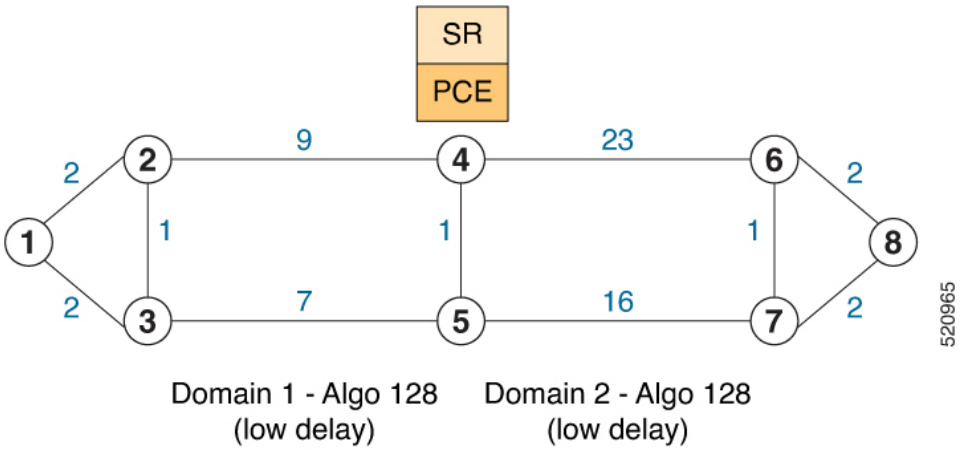
| Scenario | Domain 1 | Domain 2 |
|---|---|---|
| Scenario 1 | Flexible Algorithm 128, metric delay | Flexible Algorithm 128, metric delay |
| Scenario 2 | Flexible Algorithm 128, metric delay | Flexible Algorithm 128, metric delay, exclude affinity blue |
| Scenario 3 | Flexible Algorithm 128, metric delay, exclude affinity yellow | Flexible Algorithm 128, metric delay, exclude affinity blue |
| Scenario 4 | Flexible Algorithm 128, metric delay<br>Flexible Algorithm 129, metric IGP | Flexible Algorithm 128, metric delay<br>Flexible Algorithm 129, metric IGP |

**Note** The use of a Flexible Algorithm constraint in a multi-domain SR topology does not preclude the use of an SR policy that are optimized for a particular metric type. For example, a policy can request a PCE for a Multi Domain policy based on metric delay. SR-PCE computes the path and encodes it with regular prefix SIDs and Adj-SIDs as required. Alternatively, a policy can request to have a constraint for a Flexible Algorithm instance X, which is defined in multiple domains and it minimizes based on metric delay. In this case, the SR-PCE computes the multi-domain path and encodes it using only Flexible Algorithm prefix SIDs. This case benefits from the optimized label stack size that Flexible Algorithm provides (1 label per domain).

# Example: SR-PCE Flexible Algorithm Multi-Domain Path Computation Use Case

The following use case depicts a multi-domain topology with two IS-IS processes, each with a Flexible Algorithm instance of 128 that minimizes metric delay. A multi-domain SR policy programmed at Node 1 leverages a Flexible Algorithm 128 path computed by the SR-PCE toward Node 8.

***Figure 21: Multi-Domain Topology***




### Configuration on Node 8

#### IS-IS and Flexible Algorithm Configuration

```
router isis 2
 is-type level-2-only
 net 49.0002.0000.0000.0008.00
 distribute link-state
 flex-algo 128
   metric-type delay
   advertise-definition

 address-family ipv4 unicast
   metric-style wide
   router-id 10.1.1.8
   segment-routing mpls
  !
 interface Loopback0
   passive
   address-family ipv4 unicast
    prefix-sid absolute 16008
    prefix-sid algorithm 128 absolute 16808
 !
```

### Configuration on Node 4 (ABR/ASBR)

#### IS-IS and Flexible Algorithm Configuration

```
router isis 1
 is-type level-2-only
 net 49.0001.0000.0000.0004.00
 distribute link-state instance-id 100
```

```
 flex-algo 128
   metric-type delay
   advertise-definition

 address-family ipv4 unicast
   metric-style wide
   router-id 10.1.1.4
   segment-routing mpls
  !
 interface Loopback0
   passive
   address-family ipv4 unicast
    prefix-sid absolute 16004
    prefix-sid algorithm 128 absolute 16804
 !
router isis 2
 is-type level-2-only
 net 49.0002.0000.0000.0004.00
 distribute link-state instance-id 200
 flex-algo 128
   metric-type delay
   advertise-definition

 address-family ipv4 unicast
   metric-style wide
   router-id 10.1.1.4
   segment-routing mpls
  !
 interface Loopback0
   passive
   address-family ipv4 unicast
    prefix-sid absolute 16004
    prefix-sid algorithm 128 absolute 16804
 !
```

### BGP-LS Configuration

```
router bgp 65000
 bgp router-id 10.1.1.4
 address-family link-state link-state
 !
 neighbor-group AS65000-LS-group
  remote-as 65000
  update-source Loopback0
  address-family link-state link-state
  !
 !
 neighbor 10.1.1.10
  use neighbor-group AS65000-LS-group
  description *** To SR-PCE ***
  !
 !
!
```

## Configuration on Node 1

### IS-IS and Flexible Algorithm Configuration

```
router isis 1
 is-type level-2-only
 net 49.0001.0000.0000.0001.00
 distribute link-state
```

```
 flex-algo 128
   metric-type delay
   advertise-definition

 address-family ipv4 unicast
   metric-style wide
   router-id 10.1.1.1
   segment-routing mpls
  !
 interface Loopback0
   passive
   address-family ipv4 unicast
    prefix-sid absolute 16001
    prefix-sid algorithm 128 absolute 16801
 !
```

### SR Policy Configuration

```
segment-routing
 traffic-eng
  policy FOO
   color 100 end-point ipv4 10.1.1.8
   candidate-paths
    preference 100
     dynamic
      pcep
      !
     !
     constraints
      segments
       sid-algorithm 128
      !
     !
    !
   !
  !
 !
!
```

### PCC Configuration

```
segment-routing
 traffic-eng
  pcc
   source-address ipv4 10.1.1.1
   pce address ipv4 10.1.1.10
    precedence 10
   !
   report-all
  !
 !
!
```

### Configuration on PCE

```
pce
 address ipv4 10.1.1.10
 rest
 !
!
router bgp 65000
 bgp router-id 10.1.1.10
 address-family link-state link-state
 !
```

```
 neighbor-group AS65000-LS-group
  remote-as 65000
  update-source Loopback0
  address-family link-state link-state
  !
 !
 neighbor 10.1.1.4
  use neighbor-group AS65000-LS-group
  description *** To Node-4 ***
  !
 !
 neighbor 10.1.1.5
  use neighbor-group AS65000-LS-group
  description *** To Node-5 ***
  !
 !
!
```

# 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 Understanding Access Lists chapter in the *IP Addresses and Services Configuration Guide for Cisco NCS 540 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.

The following example shows how to configure an ACL and apply it to the PCE:

```
pce
 address ipv4 10.1.1.5
 peer-filter ipv4 access-list sample-peer-filter
!
ipv4 access-list sample-peer-filter
 10 permit ipv4 host 10.1.1.6 any
 20 permit ipv4 host 10.1.1.7 any
 30 deny ipv4 any any
!
```

# SR-PCE IPv4 Unnumbered Interface Support

This feature allows IPv4 unnumbered interfaces to be part of an SR-PCE topology database.

An unnumbered IPv4 interface is not identified by its own unique IPv4 address. Instead, it is identified by the router ID of the node where this interfaces resides and the local SNMP index assigned for this interface.

This feature provides enhancements to the following components:

- IGPs (IS-IS and OSPF):
  - Support the IPv4 unnumbered interfaces in the SR-TE context by flooding the necessary interface information in the topology
- SR-PCE:

**Note** SR-PCE and path computation clients (PCCs) need to be running Cisco IOS XR 7.0.2 or later.

  - Compute and return paths from a topology containing IPv4 unnumbered interfaces.
  - Process reported SR policies from a head-end router that contain hops with IPv4 unnumbered adjacencies.

    PCEP extensions for IPv4 unnumbered interfaces adhere to IETF RFC8664 “PCEP Extensions for Segment Routing” (https://datatracker.ietf.org/doc/rfc8664/). The unnumbered hops use a Node or Adjacency Identifier (NAI) of type 5. This indicates that the segment in the explicit routing object (ERO) is an unnumbered adjacency with an IPv4 ID and an interface index.
- SR-TE process at the head-end router:
  - Compute its own local path over a topology, including unnumbered interfaces.
  - Process PCE-computed paths that contain hops with IPv4 unnumbered interfaces.
  - Report a path that contains hops with IPv4 unnumbered interfaces to the PCE.

### Configuration Example

The following example shows how to configure an IPv4 unnumbered interface:

```
RP/0/0/CPU0:rtrA(config)# interface GigabitEthernet0/0/0/0
RP/0/0/CPU0:rtrA(config-if)# ipv4 point-to-point
RP/0/0/CPU0:rtrA(config-if)# ipv4 unnumbered Loopback0
```

To bring up the IPv4 unnumbered adjacency under the IGP, configure the link as point-to-point under the IGP configuration. The following example shows how to configure the link as point-to-point under the IGP configuration:

```
RP/0/0/CPU0:rtrA(config)# router ospf one
RP/0/0/CPU0:rtrA(config-ospf)# area 0
RP/0/0/CPU0:rtrA(config-ospf-ar)# interface GigabitEthernet0/0/0/0
RP/0/0/CPU0:rtrA(config-ospf-ar-if)# network point-to-point
```

### Verification

Use the **show ipv4 interface** command to display information about the interface:

```
RP/0/0/CPU0:rtrA# show ipv4 interface GigabitEthernet0/0/0/0 brief
Tue Apr  2 12:59:53.140 EDT
Interface                  IP-Address      Status                Protocol
GigabitEthernet0/0/0/0     192.168.0.1     Up                    Up
```

This interface shows the IPv4 address of Loopback0.

Use the **show snmp interface** command to find the SNMP index for this interface:

```
RP/0/0/CPU0:rtrA# show snmp interface
Tue Apr  2 13:02:49.190 EDT
ifName : Null0                 ifIndex : 3
ifName : Loopback0             ifIndex : 10
ifName : GigabitEthernet0/0/0/0  ifIndex : 6
```

The interface is identified with the pair (IPv4:192.168.0.1, index:6).

Use the **show ospf neighbor** command to display the adjacency:

```
RP/0/0/CPU0:rtrA# show ospf neighbor gigabitEthernet 0/0/0/0 detail
…
Neighbor 192.168.0.4, interface address 192.168.0.4
    In the area 0 via interface GigabitEthernet0/0/0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    …
    Adjacency SIDs:
        Label: 24001,    Dynamic, Unprotected
    Neighbor Interface ID: 4
```

The output of the **show pce ipv4 topology** command is enhanced to display the interface index instead of the IP address for unnumbered interfaces:

```
RP/0/0/CPU0:sr-pce# show pce ipv4 topology
…
  Link[2]: unnumbered local index 6, remote index 4
    Local node:
      OSPF router ID: 192.168.0.1 area ID: 0 ASN: 0
    Remote node:
      TE router ID: 192.168.0.4
      OSPF router ID: 192.168.0.4 area ID: 0 ASN: 0
    Metric: IGP 1, TE 1, Latency 1 microseconds
    Bandwidth: Total 125000000 Bps, Reservable 0 Bps
    Admin-groups: 0x00000000
    Adj SID: 24001 (unprotected)
```

The output of **show pce lsp detail** command includes unnumbered hops:

```
RP/0/0/CPU0:sr-pce# show pce lsp detail
…
   Reported path:
     Metric type: TE, Accumulated Metric 3
      SID[0]: Adj unnumbered, Label 24001, local 192.168.0.1(6), remote 192.168.0.4(4)
      SID[1]: Adj unnumbered, Label 24002, local 192.168.0.4(7), remote 192.168.0.3(7)
      SID[2]: Adj unnumbered, Label 24000, local 192.168.0.3(5), remote 192.168.0.2(5)
   Computed path: (Local PCE)
     Computed Time: Wed Apr 03 11:01:46 EDT 2019 (00:01:06 ago)
     Metric type: TE, Accumulated Metric 3
      SID[0]: Adj unnumbered, Label 24001, local 192.168.0.1(6), remote 192.168.0.4(4)
      SID[1]: Adj unnumbered, Label 24002, local 192.168.0.4(7), remote 192.168.0.3(7)
      SID[2]: Adj unnumbered, Label 24000, local 192.168.0.3(5), remote 192.168.0.2(5)
```

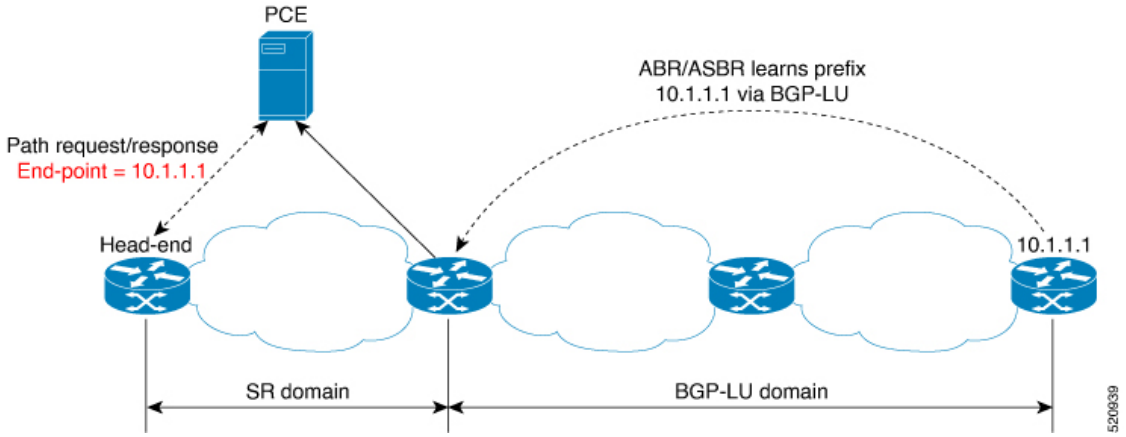
# Inter-Domain Path Computation Using Redistributed SID

A Path Computation Element (PCE) computes SR-TE paths based on SR topology database that stores connectivity, state, and TE attributes of SR network nodes and links. BGP Labeled Unicast (BGP-LU) provides MPLS transport across IGP boundaries by advertising loopbacks and label binding of impact edge and border routers across IGP boundaries.

This feature adds new functionality to the SR-PCE that enables it to compute a path for remote non-SR end-point device distributed by BGP-LU.

The remote end-point device in the BGP-LU domain is unknown to the SR-PCE. For the SR-PCE to know about the end-point device, the gateway ABR/ASBR learns the end-point prefix via BGP-LU. The prefix is then redistributed to SR-PCE topology database from the gateway ABR/ASBR. SR-PCE then can compute the best path from the head-end device to the selected gateway router.

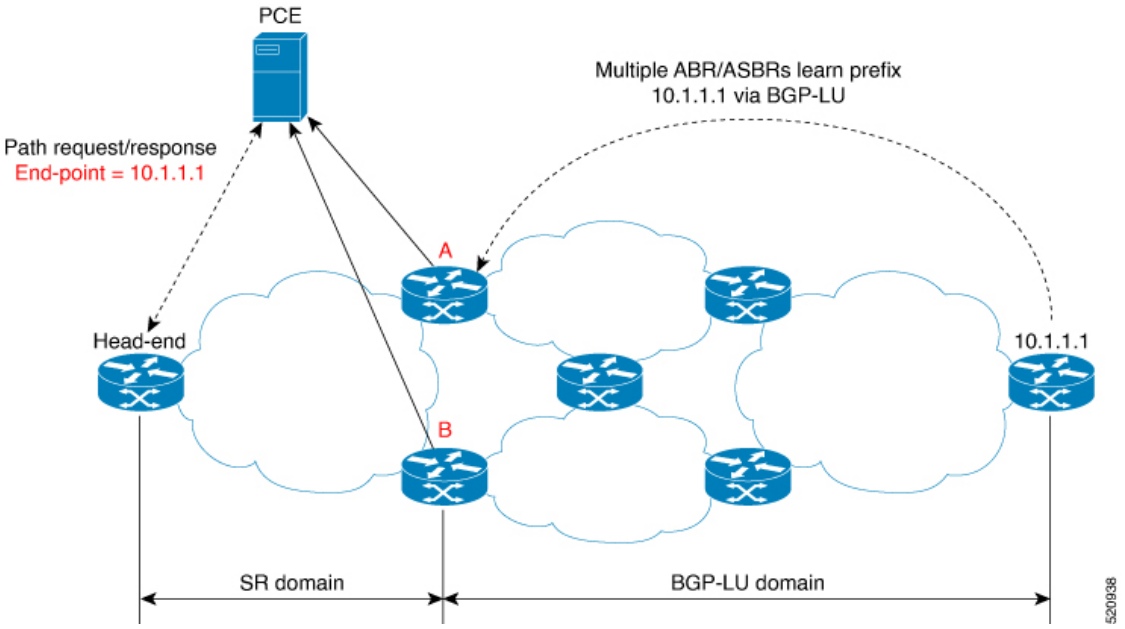
The following topology shows an SR domain and a BGP-LU domain, with a gateway ABR/ASBR between the two domains.




1. The gateway ABR/ASBR is configured with BGP/IGP helper to learn the remote prefix through BGP-LU and redistribute the remote prefix to the IGP helper, then to SR-PCE.
2. The SR-PCE selects the best gateway node to BGP-LU domain and computes the path to reach the remote prefix through the gateway node.
3. The head-end device in the SR domain requests a path to the remote destination and signals the SR profile interworking with the BGP-LU domain.

The BGP-LU prefix advertisement to SR-PCE Traffic Engineer Database (TED) is done by creating an IGP helper on the ABR/ASBR to redistribute BGP-LU prefix information to IGP. IGP then sends the prefix information to the SR-PCE via BGP-LS.

If there are multiple ABR/ASBRs advertising the same remote BGP-LU prefix, the SR-PCE selects the best gateway node to the BGP-LU domain using the accumulative metric from the head-end device to the gateway and the advertised metric from the gateway to the destination.




# Example: Inter-Domain Path Computation Using Redistributed SID

The following examples show the configurations for the IGP helper, BGP-LU, and proxy BGP-SR:

### Configuration on the End-Point Device

Configure the end-point device to allocate a label for the BGP-LU prefix on the end-point device:

```
router bgp 3107
bgp router-id 1.0.0.8
address-family ipv4 unicast
  network 1.0.0.8/32 route-policy bgplu-com
  allocate-label all

route-policy bgplu-com
  set community (65002:999)
end-policy
```

### Configuration on the Gateway ABR/ASBR

1. Configure the remote prefix set and create the route policy for the BGP-LU domain:

```
prefix-set bgplu
  1.0.0.7/32,
  1.0.0.8/32,
  1.0.0.101/32,
  1.0.0.102/32
end-set
!

route-policy bgp2isis
  if destination in bgplu then
    pass
  else
    drop
  endif
```

```
end-policy
!
end
```

2. Configure the helper IGP instance on the Loopback interface:

```
router isis 101
 is-type level-2-only
 net 49.0001.0000.1010.1010.00
 distribute link-state instance-id 9999
 nsf cisco
 nsf lifetime 120
 address-family ipv4 unicast
  metric-style wide
  maximum-paths 64
  router-id Loopback10
  redistribute bgp 3107 metric 200 route-policy bgp2isis
  segment-routing mpls sr-prefer
!
interface Loopback10 >>> this loopback is for gateway SR-TE node-id
  passive
  address-family ipv4 unicast
   prefix-sid index 2001 explicit-null
```

3. Configure the gateway proxy BGP-SR and SR Mapping Server to allocate SR labels:

```
router bgp 3107
 address-family ipv4 unicast
  segment-routing prefix-sid-map
  allocate-label all

segment-routing
global-block 16000 23999
mapping-server
  prefix-sid-map
   address-family ipv4
    1.0.0.7/32 2007
    1.0.0.8/32 2008
    1.0.0.101/32 2101
    1.0.0.102/32 2102
```

# PCE Support for MPLS-TE LSPs

This feature allows Cisco's SR-PCE to act as a Path Computation Element (PCE) for MPLS Traffic Engineering Label Switched Paths (MPLS-TE LSPs).

**Note** For more information about MPLS-TE, refer to the "Implementing MPLS Traffic Engineering" chapter in the *MPLS Configuration Guide for Cisco NCS 540 Series Routers*.

The supported functionality is summarized below:

- PCE type: Active Stateful PCE
- MPLS-TE LSP initiation methods:
  - PCE Initiated—An active stateful PCE initiates an LSP and maintains the responsibility of updating the LSP.

- PCC Initiated—A PCC initiates the LSP and may delegate the control later to the Active stateful PCE.

- MPLS-TE LSP metric—Metric optimized by the path computation algorithm:
  - IGP metric
  - TE metric
  - Latency metric

- MPLS-TE LSP constraints—TE LSP attributes to be taken into account by the PCE during path computation:
  - Resource Affinities
  - Path Disjointness

- MPLS-TE LSP parameters:
  - Setup priority—The priority of the TE LSP with respect to taking resources
  - Hold priority—The priority of the TE LSP with respect to holding resources
  - FRR L flag—The "Local Protection Desired" bit. Can be set from an application instantiating an MPLS-TE LSP via SR-PCE. SR-PCE passes this flag to the PCC, and the PCC will enable FRR for that LSP.
  - Signaled Bandwidth—This value can be set from an application instantiating an MPLS-TE LSP via SR-PCE. SR-PCE passes this value to the PCC.
  - Binding SID—A segment identifier (SID) that a headend binds to an MPLS-TE LSP. When the headend receives a packet with active segment (top MPLS label) matching the BSID of a local MPLS-TE LSP, the headend steers the packet into the associated MPLS-TE LSP.

Cisco Crosswork Optimization Engine is an application that leverages the SR-PCE in order to visualize and instantiate MPLS-TE LSPs. For more information, refer to the Visualize SR Policies and RSVP-TE Tunnels chapter in the Cisco Crosswork Optimization Engine 1.2.1 User Guide.

**Note** No extra configuration is required to enable MPLS-TE support at SR-PCE.

### Example: Configuring a PCEP Session (Stateful Mode) on MPLS-TE PCC

The following example shows the configuration for an MPLS-TE PCC to establish a PCEP session with a PCE (IPv4 address 10.1.1.100).

**Note** MPLS-TE PCC must operate in the stateful PCEP mode when connecting to SR-PCE.

The **instantiation** keyword enables the PCC to support MPLS-TE LSP instantiation by PCE (PCE-initiated).

The **report** keyword enables the PCC to report all the MPLS-TE LSPs configured on that node.

**Note** PCE-initiated LSPs are automatically reported to all configured PCEs.

The **autoroute-announce** keyword enables autoroute-announce globally for all PCE-initiated LSPs on the PCC.

The **redundancy pcc-centric** keywords enable PCC-centric high-availability model for PCE-initiated LSPs. The PCC-centric model changes the default PCC delegation behavior to the following:

- After LSP creation, LSP is automatically delegated to the PCE that computed it.
- If this PCE is disconnected, then the LSP is redelegated to another PCE.
- If the original PCE is reconnected, then the delegation fallback timer is started. When the timer expires, the LSP is redelegated back to the original PCE, even if it has worse preference than the current PCE.

```
mpls traffic-eng
 pce
  peer ipv4 10.1.1.100
  !
  stateful-client
   instantiation
   report
   autoroute-announce
   redundancy pcc-centric
  !
 !
!
end
```

### Example: Configuring Multiple PCEP Sessions from a PCC Acting as MPLS-TE and SR-TE Headend Toward a Common PCE

The following example shows the configuration for a PCC (IPv4 addresses 10.1.1.1 and 10.1.1.2) to establish two PCEP sessions with a common PCE (IPv4 address 10.1.1.100). One session is configured under MPLS-TE, and the other under SR-TE.

**Note** The two PCEP sessions must use a different source address on the PCC when connecting to the same PCE.

For more information regarding PCEP configuration at SR-TE PCC, see the *Configure the Head-End Router as PCEP PCC* topic.

```
mpls traffic-eng
 pce
  peer source ipv4 10.1.1.1
  peer ipv4 10.1.1.100
  !
 !
!
end

segment-routing
 traffic-eng
```

```
  pcc
   source-address ipv4 10.1.1.2
   pce address ipv4 10.1.1.100
   !
  !
 !
!
end
```

# Configuring the North-Bound API on SR-PCE

*Table 31: Feature History Table*

*Table 32: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR-PCE: Stateful North-Bound API for Tree-SID | Release 7.5.1 | The SR-PCE provides a north-bound HTTP-based API to allow communication between the SR-PCE and the Cisco Crosswork Optimization Engine.<br>This release adds stateful north-bound APIs to support real-time monitoring of Tree-SID states on the SR-PCE using a subscription model.<br>For more information, refer to the Cisco Crosswork Optimization Engine User Guides. |

*Table 33: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR-PCE: North-Bound API for SRv6 and Flexible Algorithm in Cisco Optimization Engine (COE) v3.0 release | Release 7.3.2 | The SR-PCE provides a north-bound HTTP-based API to allow communication between the SR-PCE and the Cisco Crosswork Optimization Engine.<br>This release adds support for the following:<br>• Reporting of Flexible Algorithm participation and definitions<br>• SRv6 topology information (nodes, links, Node uSIDs and Adj uSIDs)<br>• SRv6 uSID list and uB6 SIDs allocated for a policy<br>For more information, refer to the Cisco Crosswork Optimization Engine User Guides. |

The SR-PCE provides a north-bound HTTP-based API to allow communication between SR-PCE and external clients and applications.

Over this API, an external application can leverage the SR-PCE for topology discovery, SR policy discovery, and SR policy instantiation.

The Cisco Crosswork Optimization Engine is an application that leverages the SR-PCE. For more information, refer to the Cisco Crosswork Optimization Engine User Guides.

Use the following commands under PCE configuration mode to configure the API to allow communication between SR-PCE and external clients or applications.

**Note** The API server is enabled by default when SR-PCE is configured.

| Command | Description |
|---|---|
| **rest authentication basic** | (Optional) Specify basic (plaintext) authentication. By default, authentication is disabled. |
| **rest username** *password* {**clear** \| **encrypted**} *password* | Add credentials when connecting to API.<br>**Note** This command is used only if authentication is configured. |

| Command | Description |
|---|---|
| **rest sibling ipv4** *address* | Opens a synchronization channel to another PCE in the same high availability (HA) pair.<br>**Note** For more information regarding SR-PCE HA pairs, refer to the Multiple Cisco SR-PCE HA Pairs chapter of the Cisco Crosswork Optimization Engine 1.2.1 User Guide. |

| Command | Description |
|---|---|
| **api authentication** {**basic** \| **digest**} | Specify the type of authentication:<br>• **basic** – Use HTTP Basic authentication (plaintext)<br>• **digest** – Use HTTP Digest authentication (MD5) |
| **api username** *password* {**clear** \| **encrypted**} *password* | Add credentials when connecting to API. |
| **api sibling ipv4** *address* | Opens a synchronization channel to another PCE in the same high availability (HA) pair.<br>**Note** For more information regarding SR-PCE HA pairs, refer to the Multiple Cisco SR-PCE HA Pairs chapter of the Cisco Crosswork Optimization Engine 1.2.1 User Guide. |

### Example: Configuring API on SR-PCE

```
pce
 address ipv4 10.1.1.100
 rest
  user admin
   password encrypted 1304131F0202
  !
  authentication basic
  sibling ipv4 10.1.1.200
 !
!
end
```

```
pce
 address ipv4 10.1.1.100
 api
  user admin
   password encrypted 1304131F0202
  !
  authentication digest
  sibling ipv4 10.1.1.200
 !
```

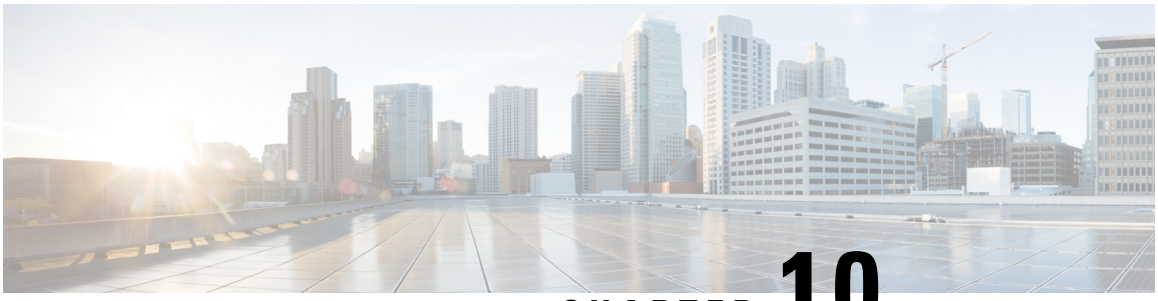
```
!
end
```

The following example shows the current active connections:

```
RP/0/0/CPU0:pce1# show tcp brief | i 8080
Thu Aug  6 00:40:15.408 PDT
0xe9806fb8 0x60000000      0      0  :::8080               :::0                  LISTEN
0xe94023b8 0x60000000      0      0  10.1.1.100:50487      10.1.1.200:8080       ESTAB
0xeb20bb40 0x60000000      0      0  10.1.1.100:8080       10.1.1.200:44401      ESTAB
0xe98031a0 0x60000000      0      0  0.0.0.0:8080          0.0.0.0:0             LISTEN
```

The first and fourth entries show the API server listening for IPv4 and IPv6 connections.

The second and third entries show the established sibling connection between PCE1 (10.1.1.100) and PCE2 (10.1.1.200).



CHAPTER 10

# Configure Performance Measurement

Network performance metrics is a critical measure for traffic engineering (TE) in service provider networks. Network performance metrics include the following:

- Packet loss
- Delay
- Delay variation
- Bandwidth utilization

These network performance metrics provide network operators information about the performance characteristics of their networks for performance evaluation and help to ensure compliance with service level agreements. The service-level agreements (SLAs) of service providers depend on the ability to measure and monitor these network performance metrics. Network operators can use Segment Routing Performance Measurement (SR-PM) feature to monitor the network metrics for links and end-to-end TE label switched paths (LSPs).

The following table explains the functionalities supported by performance measurement feature for measuring delay for links or SR policies.

***Table 34: Performance Measurement Functionalities***

| Functionality | Details |
|---|---|
| Profiles | You can configure different default profiles for different types of delay measurements. Use the "interfaces" delay profile type for link-delay measurement. The "sr-policy" delay profile type is used for SR policy delay measurements. Delay profile allows you to schedule probe and configure metric advertisement parameters for delay measurement. |
| Protocols | |
| Probe and burst scheduling | Schedule probes and configure metric advertisement parameters for delay measurement. |
| Metric advertisements | Advertise measured metrics periodically using configured thresholds. Also supports accelerated advertisements using configured thresholds. |
| Measurement history and counters | Maintain packet delay and loss measurement history, session counters, and packet advertisement counters. |

### Usage Guidelines and Limitations

Performance Measurement (PM) probes typically follow the designated Segment Routing Traffic Engineering (SR-TE) path. However, in certain scenarios, the convergence of the PM probes and the SR-TE path may occur at different times. During this convergence period, PM probes may temporarily follow the IGP path and utilize an alternate egress interface until full convergence is achieved.


- Liveness Monitoring, on page 322
- Delay Measurement, on page 333


# Liveness Monitoring

Liveness refers to the ability of the network to confirm that a specific path, segment, or a node is operational and capable of forwarding packets. Liveness checks are essential for maintaining network availability and reliability. See *Configure PTP* in *System Management Configuration Guide* for more information on configuring PTP.

### Benefits

- **Fault Detection**: You can quickly identify if a device is down, which allows for immediate response and troubleshooting.
- **Load Balancing**: You can identify if the devices in a network are live, so work can be distributed more evenly across the network, preventing overloading of specific components and improving overall performance.
- **System Health**: You can provide an ongoing snapshot of a system's health, helping to identify potential issues before they become significant problems.
- **Maintenance Planning**: Liveness information can also help with maintenance planning, as system administrators can understand which components are live or down and plan maintenance and downtime accordingly without significant disruption to services.
- **Security**: Regular liveness checks can also play a role in maintaining network security. Administrators can take proactive steps to mitigate the damage and prevent future incidents by identifying unusual activity that might indicate a security breach or attack.

You can determine liveness for SR Policy and IP Endpoint.

# IP Endpoint Liveness Monitoring

***Table 35: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| IP Endpoint Delay Measurement and Liveness Monitoring | Release 7.4.1 | This feature measures the end-to-end delay and monitors liveness of a specified IP endpoint node, including VRF-aware (awareness of multiple customers belonging to different VRFs).<br>This feature is supported on IPv4, IPv6, and MPLS data planes. |

The Segment Routing Performance Measurement (SR-PM) for IP endpoint liveness is a type of node liveness that involves testing whether an IP endpoint or a device identified by an IP address is available to send and receive data.

IP endpoint liveness is verified by sending a request to the IP address of the endpoint and waiting for a response. The probe could be an ICMP echo request (Ping), a TCP packet, a UDP packet, or any other type of packet that the endpoint would respond to.

- If a response is received, the endpoint is considered *live*.
- If no response is received within a certain time frame, the endpoint is considered *down* or *unreachable*.

IP endpoint dynamically measures the liveness towards a specified IP endpoint. IP endpoints can be located in a default or nondefault VRFs. IP endpoint is any device in the network a device identified by an IP address.

Liveness of an IP endpoint is verified by sending a request to the IP address of the endpoint and waiting for a response, which is referred to as a probe.

The endpoint of a probe is defined by an IP address, which can be either IPv4 or IPv6. This IP address can be any address that the sender can reach, such as a local interface or a remote node or host, either within an operator's network or accessible via a VRF.

The endpoint of a probe can be any IP address reachable by the sender. For example, a local interface or a remote node or host located within an operator's network or reachable through a VRF.

The IP address of the endpoint can be reached through an IP path, MPLS, LSP , or IP tunnel (GRE).

- When the endpoint is reachable using an MPLS LSP (for example, SR, LDP, RSVP-TE, SR Policy), the forwarding stage imposes the corresponding MPLS transport labels.
- When the endpoint is reachable via a GRE tunnel, the forwarding stage imposes the corresponding GRE header.
- When the endpoint is reachable via a VRF in an MPLS network, the forwarding stage imposes the corresponding MPLS service labels. In the forward path, the sender node uses the configured VRF for the endpoint address. In the return path, the reflector node derives the VRF based on which incoming VRF label the probe packet is received with.

You can configure the following parameters in the **performance-measurement** command:

- **Endpoint**: The endpoint of a probe is defined by an IP address, which can be either IPv4 or IPv6. This IP address can be any address that the sender can reach, such as a local interface or a remote node or host, either within an operator's network or accessible via a VRF.

  The endpoint of a probe can be any IP address reachable by the sender. For example, a local interface or a remote node or host located within an operator's network or reachable through a VRF.

  Use the **performance-measurement endpoint** command to configure a probe endpoint source and destination addresses on a sender node.

- **VRF**: You can define the endpoint point IP address belonging to a specific VRF. Use the **performance-measurement endpoint {ipv4 | ipv6} ip_addr [vrf WORD]** command to configure an endpoint to define the VRF. Endpoint segment list configuration is not supported under nondefault VRF.
  - VRF-awareness allows operators to deploy probes in the following scenarios:
    - Managed Customer Equipment (CE) scenarios:
      - PE to CE probes

    - CE to CE probes

- Unmanaged Customer Equipment (CE) scenarios:
    - PE to PE probes
    - PE to PE (source from PE-CE interface) probes

- **Source address**: You can define the source of the endpoint using the endpoint specific source address and the global source address.

  Global source address configuration is applied to all the endpoints when the endpoint specific source address configuration isn't specified. endpoint specific configuration overrides all the global source address configuration for those specific endpoints for which source addresses are configured.

  For Micro-SID configuration for IPv4 endpoint sessions, if IPv6 global source address is configured, then it applies the configured global IPv6 source address for the IPv6 header in the SRv6 packet. If IPv6 global address is not configured, then It does not form a valid SRv6 packet.

  You can use the **source-address** keyword under the **performance-measurement** command to define the global source address or use the keyword under **performance-measurement endpoint** to define endpoint specific source address.

### Usage Guidelines and Limitations

- Liveness session without segment list for an endpoint in a non-default VRF is not supported.
- SR Performance Measurement endpoint session over BVI interface is not supported.

## IP Endpoint Liveness Detection in an SR MPLS Network

IP endpoint liveness detection leverages the loopback measurement-mode. The following workflow describes the sequence of events.

1. The sender creates and transmits the PM probe packets.

   The IP destination address (DA) on the probe packets is set to the loopback value of the sender itself.

   The transmit timestamp (T1) is added to the payload.

   The probe packet is encapsulated with the label corresponding to the endpoint.

2. The network delivers the PM probe packets following the LSP toward the endpoint.

3. The end-point receives the PM probe packets.

   Packets are forwarded back to the sender based on the forwarding entry associated with the IP DA of the PM probe packet. If an LSP exists, the probe packet is encapsulated with the label of the sender.

4. The sender node receives the PM probe packets.

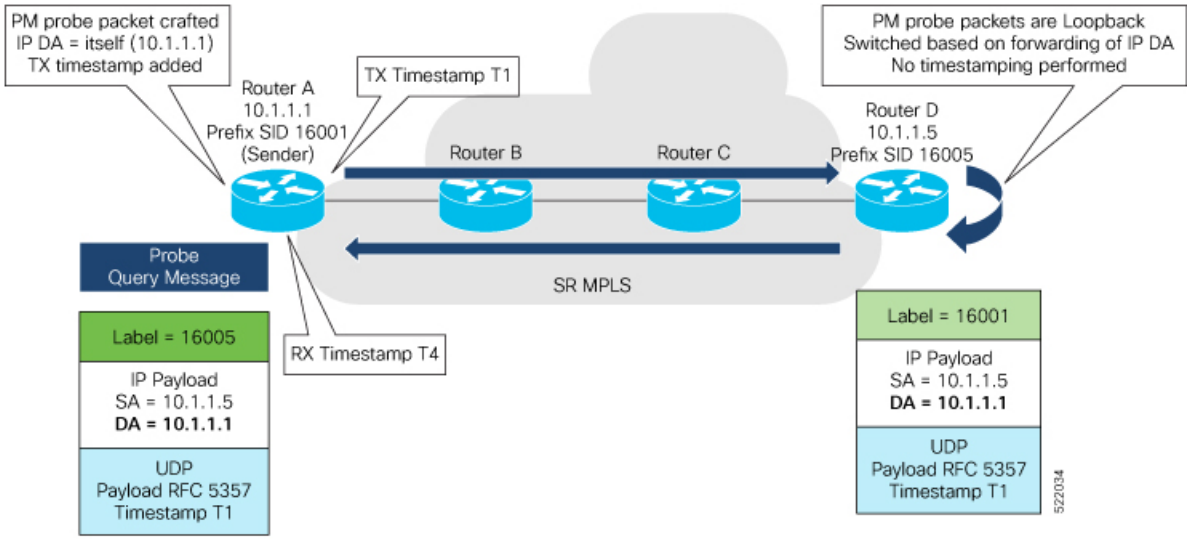
   The received timestamp (T4) stored.

   If the sender node doesn't receive the specified number of probe packets (based on the configured multiplier), the sender node declares the PM session as down.

The following figure illustrates a liveness detection probe toward an IP endpoint learned by the IGP. The network interconnecting the sender and reflector provides MPLS connectivity with Segment Routing.

The liveness detection multiplier is set to 5 to specify the number of consecutive missed probe packets before the PM session is declared as down.

***Figure 22: IP Endpoint Liveness Detection***




## Configuration Example

```
RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 10.1.1.5
RouterA(config-pm-ep)# source-address ipv4 10.1.1.1
RouterA(config-pm-ep)# liveness-detection
RouterA(config-pm-ep-ld)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# liveness-profile endpoint default
RouterA(config-pm-ld-ep)# liveness-detection
RouterA(config-pm-ld-ep-ld)# multiplier 5
RouterA(config-pm-ld-ep-ld)# exit
RouterA(config-pm-ld-ep)# probe
RouterA(config-pm-ld-ep-probe)# measurement-mode loopback
```

## Running Configuration

```
performance-measurement
 endpoint ipv4 10.1.1.5
  source-address ipv4 10.1.1.1
  liveness-detection
  !
 !
 liveness-profile endpoint default
  liveness-detection
   multiplier 5
  !
  probe
   measurement-mode loopback
  !
 !
```

```
!
end
```

## Verification

```
RouterA# show performance-measurement endpoint ipv4 10.1.1.5

--------------------------------------------------------------------------------
0/RSP0/CPU0
--------------------------------------------------------------------------------

Endpoint name: IPv4-10.1.1.5-vrf-default
  Source address              : 10.1.1.1
  VRF name                    : default
  Liveness Detection          : Enabled
  Profile Keys:
    Profile name              : default
    Profile type              : Endpoint Liveness Detection

  Segment-list                : None
  Session State: Down
  Missed count: 0
```

# SR Policy Liveness Monitoring

*Table 36: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR Performance Measurement Named Profiles | Release 7.3.1 | You can use this feature to create specific performance measurement delay and liveness profiles, and associate it with an SR policy.<br>This way, a delay or liveness profile can be associated with a policy for which the performance measurement probes are enabled, and performance measurement is precise, and enhanced.<br>The **performance-measurement delay-profile sr-policy** command was updated with the **name** *profile* keyword-argument combination.<br>The **performance-measurement liveness-profile sr-policy** command was updated with the **name** *profile* keyword-argument combination.<br>The **performance-measurement delay-measurement** command was updated with **delay-profile name** *profile*.<br>The **performance-measurement liveness-detection** command was updated with **liveness-profile name** *profile* |
| SR Policy Liveness Monitoring | Release 7.3.1 | This feature allows you to verify end-to-end traffic forwarding over an SR Policy candidate path by periodically sending performance monitoring packets. |

SR Policy liveness monitoring allows you to verify end-to-end traffic forwarding over an SR Policy candidate path by periodically sending performance monitoring (PM) packets. The head-end router sends PM packets to the SR policy's endpoint router, which sends them back to the head-end without any control-plane dependency on the endpoint router.

The following are benefits to using SR-PM liveness monitoring:

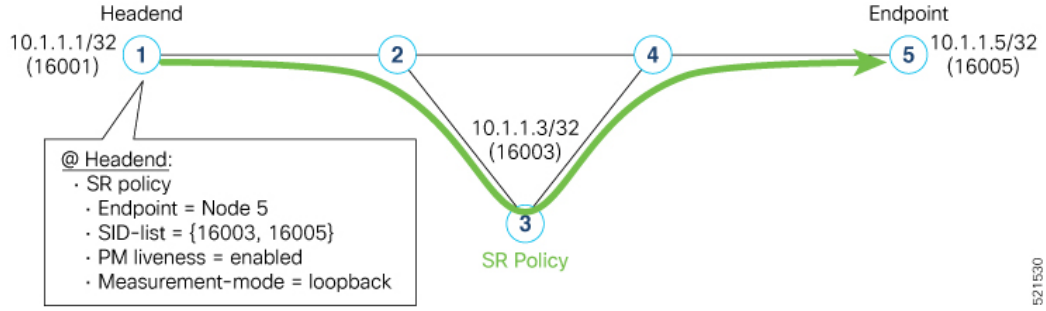
- Allows both liveness monitoring and delay measurement using a single-set of PM packets as opposed to running separate monitoring sessions for each purpose. This improves the overall scale by reducing the number of PM sessions required.

- Eliminates network and device complexity by reducing the number of monitoring protocols on the network (for example, no need for Bidirectional Failure Detection [BFD]). It also simplifies the network and device operations by not requiring any signaling to bootstrap the performance monitoring session.
- Improves interoperability with third-party nodes because signaling protocols aren't required. In addition, it leverages the commonly supported TWAMP protocol for packet encoding.
- Improves liveness detection time because PM packets aren't punted on remote nodes
- Provides a common solution that applies to data-planes besides MPLS, including IPv4, IPv6, and SRv6.

### How it works?

The workflow associated with liveness detection over SR policy is described in the following sequence.

Consider an SR policy programmed at head-end node router 1 towards end-point node router 5. This SR policy is enabled for liveness detection using the loopback measurement-mode.




- **A**: The head-end node creates and transmits the PM probe packets.

  The IP destination address (DA) on the probe packets is set to the loopback value of the head-end node itself.

  A transmit (Tx) timestamp is added to the payload.

  Optionally, the head-end node may also insert extra encapsulation (labels) to enforce the reverse path at the endpoint node.

  Finally, the packet is injected into the data-plane using the same encapsulation (label stack) of that of the SR policy being monitored.

- **B**: The network delivers the PM probe packets as it would user traffic over the SR policy.

- **C**: The end-point node receives the PM probe packets.

  Packets are switched back based on the forwarding entry associated with the IP DA of the packet. This would typically translate to the end-point node pushing the prefix SID label associated with the head-end node.

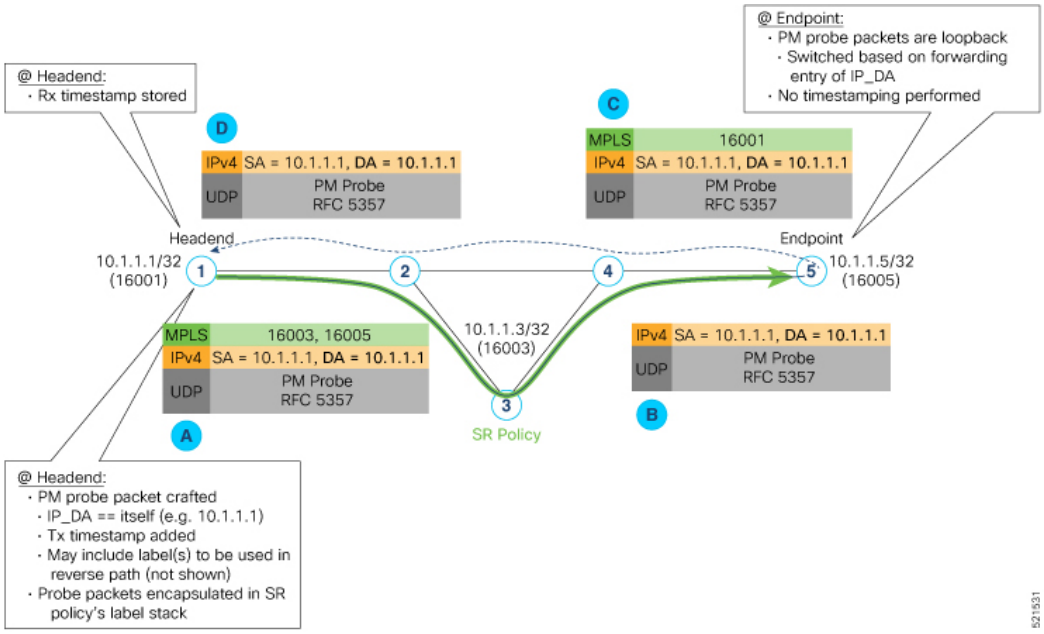
  If the head-end node inserted label(s) for the reverse path, then the packets are switched back at the end-point node based on the forwarding entry associated with the top-most reverse path label.

- **D**: Headend node receives the PM probe packets.

  A received (Rx) timestamp stored.

  If the head-end node receives the PM probe packets, the head-end node assume that the SR policy active candidate path is up and working.

If the head-end node doesn't receive the specified number of consecutive probe packets (based on configured multiplier), the head-end node assumes the candidate path is down and a configured action is trigerred.




### Usage Guidelines and Limitations

The following usage guidelines and limitations apply:

- SR-PM liveness-detection over SR Policy is supported on manually configured SR Policies and On-Demand SR Policies (ODN).
- SR-PM liveness-detection over SR Policy is not supported on PCE-initiated SR Policies.
- SR-PM liveness-detection and delay-measurement aren't supported together
- When liveness-profile isn't configured, SR Policies use the default values for the liveness-detection profile parameters.

## Configure SR Policy Liveness Monitoring in an MPLS Network

Configuring SR Policy liveness monitoring involves the following steps:

- Configuring a performance measurement liveness profile to customize generic probe parameters
- Enabling liveness monitoring under SR Policy by associating a liveness profile, and customizing SR policy-specific probe parameters

Liveness monitoring parameters are configured under **performance-measurement liveness-profile** sub-mode. The following parameters are configurable:

- **liveness-profile sr-policy {default | name** *name*}

  Parameters defined under the **sr-policy default** liveneness-profile apply to any SR policy with liveness monitoring enabled and that does not reference a non-default (named) liveneness-profile.

- **probe**: Configure the probe parameters.
- **measurement-mode**: Liveness detection must use loopback mode (see Measurement Modes, on page 334).
- **burst interval**: Interval for sending probe packet. The default value is 3000 milliseconds and the range is from 30 to 15000 milliseconds.
- **tos dscp** *value*: The default value is 48 and the range is from 0 to 63. You can modify the DSCP value of the probe packets, and use this value to priortize the probe packets from headend to tailend.
- **sweep destination ipv4 127.*x*.*x*.*x* range** *range*: Configure SR Policy ECMP IP-hashing mode. Specifiy the number of IP addresses to sweep. The range is from 0 (default, no sweeping) to 128. The option is applicable to IPv4 packets.

**Note** The destination IPv4 headendaddress 127.x.x.x – 127.y.y.y is used in the Probe messages to take advantages of 3-tuple IP hashing (source-address, destination-address, and local router ID) for ECMP paths of SR-MPLS Policy.

The destination IPv4 address must be 127/8 range (loopback), otherwise it will be rejected.

**Note** One PM session is always created for the actual endpoint address of the SR Policy.

- **liveness-detection**: Configure the liveness-detection parameters:
- **multiplier**: Number of consecutive missed probe packets before the PM session is declared as down. The range is from 2 to 10, and the default is 3.

**Note** The detection-interval is equal to (burst-interval * multiplier).

### Enabling Liveness Monitoring under SR Policy

Enable liveness monitoring under SR Policy, associate a liveness-profile, and configure SR Policy-specific probe parameters under the **segment-routing traffic-eng policy performance-measurement** sub-mode. The following parameters are configurable:

- **liveness-detection**: Enables end-to-end SR Policy Liveness Detection for all segment-lists of the active and standby candidate-path that are in the forwarding table.
- **liveness-profile name** *name*: Specifies the profile name for named profiles.
- **invalidation-action {down | none}**:
  - **Down (default)**: When the PM liveness session goes down, the candidate path is immediately operationally brought down.

- **None**: When the PM liveness session goes down, no action is taken. If logging is enabled, the failure is logged but the SR Policy operational state isn't modified.

- **logging session-state-change**: Enables Syslog messages when the session state changes.
- **reverse-path label** {*BSID-value* | *NODE-SID-value*}: Specifies the MPLS label to be used for the reverse path for the reply. If you configured liveness detection with ECMP hashing, you must specify the reverse path. The default reverse path uses IP Reply.
  - *BSID-value*: The Binding SID (BSID) label for the reverse SR Policy. (This is practical for manual SR policies with a manual BSID.)
  - *NODE-SID-value*: The absolute SID label of the (local) Sender Node to be used for the reverse path for the reply.

## Configuration Examples

### Configure a Default SR-Policy PM Liveness-Profile

The following example shows a default sr-policy liveness-profile:

```
RP/0/RSP0/CPU0:ios(config)# performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)# liveness-profile sr-policy default
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# probe
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# measurement-mode loopback
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# burst-interval 1500
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tos dscp 52
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# liveness-detection
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-ld)# multiplier 5
```

Running Configuration:

```
performance-measurement
 liveness-profile sr-policy default
  liveness-detection
   multiplier 5
  !
  probe
   tos dscp 52
   measurement-mode loopback
   burst-interval 1500
  !
 !
!
end
```

### Configure a Named (Non-Default) SR-Policy PM Liveness-Profile

The following example shows a named sr-policy liveness-profile:

```
RP/0/RSP0/CPU0:ios(config)# performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)# liveness-profile name sample-profile
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# probe
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# measurement-mode loopback
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# burst-interval 1500
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tos dscp 52
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# liveness-detection
```

```
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-ld)# multiplier 5
```

### Running Configuration:

```
performance-measurement
 liveness-profile sr-policy name sample-profile
  liveness-detection
   multiplier 5
  !
  probe
   tos dscp 52
   measurement-mode loopback
   burst-interval 1500
  !
 !
!
end
```

### Configure a SR-Policy PM Liveness-Profile with Sweep Parameters

The following example shows a named liveness-profile with sweep parameters:

```
RP/0/RSP0/CPU0:ios(config)# performance-measurement
RP/0/RSP0/CPU0:ios(config-perf-meas)# liveness-profile name sample-profile
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# probe
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# measurement-mode loopback
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# burst-interval 1500
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# tos dscp 52
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# sweep
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe-sweep)# destination ipv4 127.0.0.1 range 25
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe-sweep)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-probe)# exit
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy)# liveness-detection
RP/0/RSP0/CPU0:ios(config-pm-ld-srpolicy-ld)# multiplier 5
```

Running Configuration

```
performance-measurement
 liveness-profile sr-policy name sample-profile
  liveness-detection
   multiplier 5
  !
  probe
   tos dscp 52
   sweep
    destination ipv4 127.0.0.1 range 25
   !
   measurement-mode loopback
   burst-interval 1500
  !
 !
!
end
```

### Enable Liveness Monitoring Under SR Policy

The following example shows how to enable liveness monitoring under SR Policy, associate a liveness-profile, and configure the invalidation action:

```
RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng
RP/0/RSP0/CPU0:ios(config-sr-te)# policy FOO
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# performance-measurement
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)# liveness-detection
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# liveness-profile name sample-profile
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# invalidation-action none
```

Running Config

```
segment-routing
 traffic-eng
  policy FOO
   performance-measurement
    liveness-detection
     liveness-profile name sample-profile
     invalidation-action none
    !
   !
  !
 !
!
end
```

**Enable Liveness Monitoring under SR Policy with Optional Parameters**

The following example shows how to enable liveness monitoring under SR Policy, associate a liveness-profile, and configure reverse path label and session logging:

```
RP/0/RSP0/CPU0:ios(config)# segment-routing traffic-eng
RP/0/RSP0/CPU0:ios(config-sr-te)# policy BAA
RP/0/RSP0/CPU0:ios(config-sr-te-policy)# performance-measurement
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)# liveness-detection
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# liveness-profile name sample-profile
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# invalidation-action down
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# logging session-state-change
RP/0/RSP0/CPU0:ios(config-sr-te-policy-live-detect)# exit
RP/0/RSP0/CPU0:ios(config-sr-te-policy-perf-meas)# reverse-path label 16001
```

Running Config

```
segment-routing
 traffic-eng
  policy BAA
   performance-measurement
    liveness-detection
     logging
      session-state-change
     !
     liveness-profile name sample-profile
     invalidation-action down
    !
    reverse-path
     label 16001
    !
   !
  !
 !
!
end
```

# Delay Measurement

Delay measurement is a mechanism used to measure the latency or delay experienced by data packets when they traverse a network.

The PM for delay measuremnt uses the IP/UDP packet format defined in for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy. In the case of TWAMP

Light, the Session-Reflector doesn't necessarily know about the session state. The Session-Reflector simply copies the Sequence Number of the received packet to the Sequence Number field of the reflected packet. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

### Benefits

- Network Troubleshooting: You can quickly and easily identify areas in your network with high delay and resolve network problems using delay measurement.
- Network Planning and Optimization: You can easily understand the performance of your network under various conditions and design a network that can handle expected traffic loads.
- Quality of Service (QoS): You can ensure quality of service standards are being met by continuously monitoring the delay in your network.

### Supported Delay Measurement Methods

You can measure delay using the following methods:

- Link Delay Measurement, on page 336 Use to monitor delay experienced by data packets in a single link or path between two nodes in a network.
- IP endpoint delay measurement: Use to monitor the amount of time it takes for a data packet to travel from a source device to a specific IP endpoint within a network.
- SR Policy End-to-End Delay Measurement , on page 363: Use to to monitor the end-to-end delay experienced by the traffic sent over an SR policy.

# Measurement Modes

The following table compares the different hardware and timing requirements for the measurement modes that are supported in SR PM.

***Table 37: Measurement Mode Requirements***

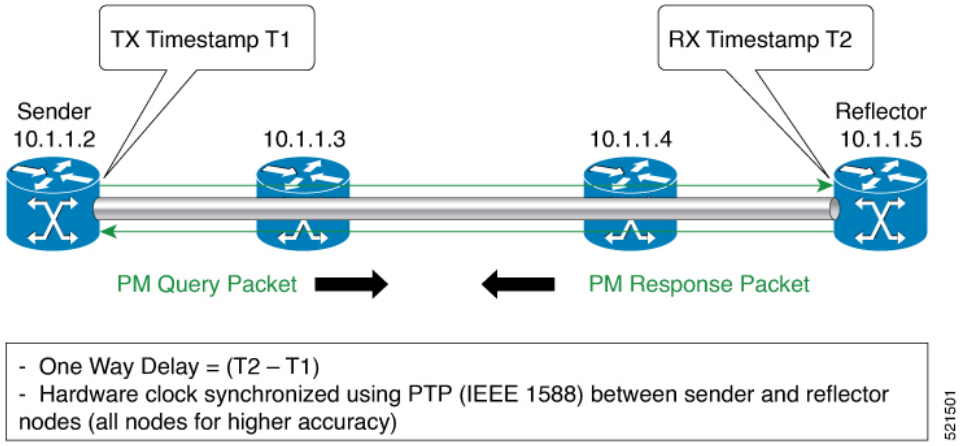
| Measurement Mode | Sender:<br>PTP-Capable HW and HW Timestamping | Reflector:<br>PTP-Capable HW and HW Timestamping | PTP Clock Synchronization between Sender and Reflector |
|---|---|---|---|
| One-way | Required | Required | Required |
| Two-way | Required | Required | Not Required |
| Loopback | Required | Not Required | Not Required |

### One-Way Measurement Mode

One-way measurement mode provides the most precise form of one-way delay measurement. PTP-capable hardware and hardware timestamping are required on both Sender and Reflector, with PTP Clock Synchronization between Sender and Reflector.

Delay measurement in one-way mode is calculated as (T2 – T1).

***Figure 23: One-Way***




The PM query and response for one-way delay measurement can be described in the following steps:

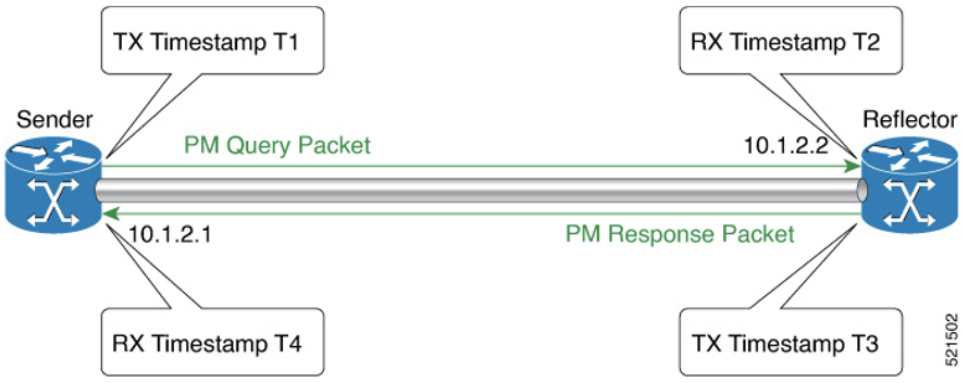
1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. The ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router.
4. One-way delay is measured using the time-stamp values in the PM packet.

### Two-Way Measurement Mode

Two-way meaurement mode provides two-way measurements. PTP-capable hardware and hardware timestamping are required on both Sender and Reflector, but PTP clock synchronization between Sender and Reflector is not required.

Delay measurement in two-way mode is calculated as ((T4 – T1) – (T3 – T2))/2

***Figure 24: Two-Way***




The PM query and response for two-way delay measurement can be described in the following steps:

1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. Ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router. The remote-end router time-stamps the packet just before sending it for two-way measurement.
4. The local-end router time-stamps the packet as soon as the packet is received for two-way measurement.
5. Delay is measured using the time-stamp values in the PM packet.

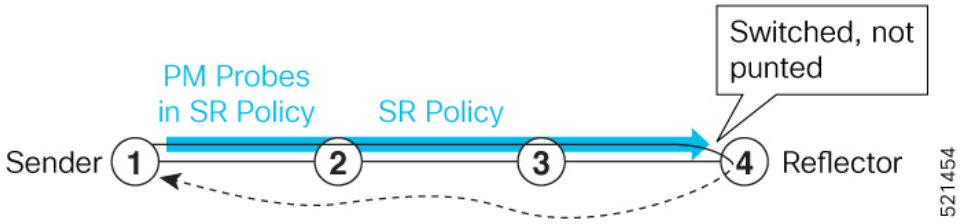
### Loopback Measurement Mode

Loopback meaurement mode provides two-way and one-way measurements. PTP-capable hardware and hardware timestamping are required on the Sender, but are not required on the Reflector.

Delay measurements in Loopback mode are calculated as follows:

- Round-Trip Delay = (T4 – T1)
- One-Way Delay = Round-Trip Delay/2

***Figure 25: Loopback***




The PM query and response for Loopback delay measurement can be described in the following steps:

1. The local-end router sends PM probe packets periodically on the SR Policy.
2. The probe packets are loopback on the endpoint node (not punted), with no timestamping on endpoint node.
3. Round-trip Delay = T4 – T1.

# Link Delay Measurement

**Note** From Cisco IOS XR Release 7.6.1 onwards, Cisco NCS 540 routers support the following features:

- Link Delay Measurement
- Named Profiles
- Static Delay Value on an Interface

*Table 38: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Link Delay Measurement with IPv6 Link Local Address | Release 7.3.1 | The performance measurement for link delay determines the source and destination IP addresses used in the OAM packet based on the IP address of the interface, where the delay measurement operation is enabled. This feature enables using the IPv6 link-local address as the OAM packet source IP address, when no IPv4 or IPv6 address is configured in the interface. |

The PM for link delay uses the IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy. In the case of TWAMP Light, the Session-Reflector doesn't necessarily know about the session state. The Session-Reflector simply copies the Sequence Number of the received packet to the Sequence Number field of the reflected packet. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

### Usage Guidelines and Restrictions for PM for Link Delay

The following restrictions and guidelines apply for the PM for link delay feature for different links.

- For broadcast links, only point-to-point (P2P) links are supported. P2P configuration on IGP is required for flooding the value.
- For link bundles, the hashing function may select a member link for forwarding but the reply may come from the remote line card on a different member link of the bundle.
- For one-way delay measurement, clocks should be synchronized on two end-point nodes of the link using PTP.
- Link delay measurement is supported on IPv4 unnumbered interfaces. An IPv4 unnumbered interface is identified by a node ID (a loopback address) and the local SNMP index assigned to the interface. Note that the reply messages could be received on any interface, since the packets are routed at the responder based on the loopback address used to identify the link.

### Configuration Example: PM for Link Delay

This example shows how to configure performance-measurement functionalities for link delay as a global default profile. The default values for the different parameters in the PM for link delay is given as follows:

- **probe measurement mode**: The default measurement mode for probe is two-way delay measurement. If you are configuring one-way delay measurement, hardware clocks must be synchronized between the local-end and remote-end routers using precision time protocol (PTP). See Measurement Modes, on page 334 for more information.
- **protocol**: Interface delay measurement using RFC 5357 with IP/UDP encap (TWAMP-Light).

- **burst interval**: Interval for sending probe packet. The default value is 3000 milliseconds and the range is from 30 to 15000 milliseconds.
- **computation interval**: Interval for metric computation. Default is 30 seconds; range is 1 to 3600 seconds.
- **periodic advertisement**: Periodic advertisement is enabled by default.
- **periodic-advertisement interval**: The default value is 120 seconds and the interval range is from 30 to 3600 seconds.
- **periodic-advertisement threshold**: Checks the minimum-delay metric change for threshold crossing for periodic advertisement. The default value is 10 percent and the range is from 0 to 100 percent.
- **periodic-advertisement minimum change**: The default value is 1000 microseconds (usec) and the range is from 0 to 100000 microseconds.
- **accelerated advertisement**: Accelerated advertisement is disabled by default.
- **accelerated-advertisement threshold**: Checks the minimum-delay metric change for threshold crossing for accelerated advertisement. The default value is 20 percent and the range is from 0 to 100 percent.
- **accelerated-advertisement minimum change**: The default value is 500 microseconds and the range is from 0 to 100000 microseconds.

```
RP/0/0/CPU0:router(config)# performance-measurement delay-profile interfaces
RP/0/0/CPU0:router(config-pm-dm-intf)# probe
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# measurement-mode one-way
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# burst-interval 60
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# computation-interval 60
RP/0/0/CPU0:router(config-pm-dm-intf-probe)# exit

RP/0/0/CPU0:router(config-pm-dm-intf)# advertisement periodic
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# interval 120
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# threshold 20
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# minimum-change 1000
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# exit

RP/0/0/CPU0:router(config-pm-dm-intf)# advertisement accelerated
RP/0/0/CPU0:router(config-pm-dm-intf-adv-acc)# threshold 30
RP/0/0/CPU0:router(config-pm-dm-intf-adv-acc)# minimum-change 1000
RP/0/0/CPU0:router(config-pm-dm-intf-adv-per)# exit
```

### Configure the UDP Destination Port

Configuring the UDP port for TWAMP-Light protocol is optional. By default, PM uses port 862 as the TWAMP-reserved UDP destination port for delay.

The UDP port is configured for each PM measurement probe type (delay, loss, protocol, authentication mode, etc.) on querier and responder nodes. If you configure a different UDP port, the UDP port for each PM measurement probe type must match on the querier and the responder nodes.

**Note** The same UDP destination port is used for delay measurement for links and SR Policy.

This example shows how to configure the UDP destination port for delay.

```
Router(config)# performance-measurement
```

```
Router(config-perf-meas)# protocol twamp-light

Router(config-pm-protocol)# measurement delay unauthenticated
Router(config-pm-proto-mode)# querier-dst-port 12000
```

### Enable PM for Link Delay Over an Interface

This example shows how to enable PM for link delay over an interface.

```
RP/0/0/CPU0:router(config)# performance-measurement
RP/0/0/CPU0:router(config-perf-meas)# interface TenGigE0/0/0/0
RP/0/0/CPU0:router(config-pm-intf)# next-hop ipv4 10.10.10.2 // Optional IPv4 or IPv6
next-hop address
RP/0/0/CPU0:router(config-pm-intf)# delay-measurement
RP/0/0/CPU0:router(config-pm-intf-dm)# exit
```

The source and destination IP addresses used in the OAM packet are determined by the IP address present on the interface where the delay-measurement operation is enabled and the setting of the optional **next-hop** address.

When the **next-hop** address is not specified, the following rules apply to determine the source and destination IP addresses used in the OAM packet:

- If an IPv4 address is configured under the interface, then:
  - OAM packet source IP address = Interface's IPv4 address
  - OAM packet destination IP address = 127.0.0.0
- Else, if an IPv6 global address is configured under the interface, then:
  - OAM packet source IP address = Interface's IPv6 global address
  - OAM packet destination IP address = 0::ff:127.0.0.0
- Else, if an IPv6 link-local address is assigned to the interface, then:
  - OAM packet source IP address = Interface's IPv6 link-local address
  - OAM packet destination IP address = 0::ff:127.0.0.0

When the **next-hop** {**ipv4** | **ipv6**} address is configured, the following rules apply to determine the source and destination IP addresses used in the OAM packet:

- If a next-hop IPv4 address is configured, then:
  - OAM packet source IP address = Interface's IPv4 address
  - OAM packet destination IP address = Configured next-hop IPv4 address

**Note** If there is no IPv4 address configured under the interface, then the delay-measurement probe does not send OAM packets.

- If a next-hop IPv6 address is configured, then:
  - OAM packet source IP address = Interface's IPv6 global address

- OAM packet destination IP address = Configured next-hop IPv6 address

**Note** If there is no IPv6 global address configured under the interface, then the delay-measurement probe does not send OAM packets.

This example shows how to enable PM for link delay over an interface with IPv4 address configured:

```
interface TenGigE0/0/0/0
 ipv4 address 10.10.10.1 255.255.255.0

performance-measurement
 interface TenGigE0/0/0/0
  delay-measurement
```

This example shows how to enable PM for link delay over an interface IPv6 address configured:

```
interface TenGigE0/0/0/0
 ipv6 address 10:10:10::1/64

performance-measurement
 interface TenGigE0/0/0/0
  delay-measurement
```

This example shows how to enable PM for link delay over an interface with a specified next-hop IPv4 address:

```
interface TenGigE0/0/0/0
 ipv4 address 10.10.10.1 255.255.255.0

performance-measurement
 interface TenGigE0/0/0/0
  next-hop ipv4 10.10.10.2
  delay-measurement
```

This example shows how to enable PM for link delay over an interface with a specified next-hop IPv6 address:

```
interface TenGigE0/0/0/0
 ipv6 address 10:10:10::1/64

performance-measurement
 interface TenGigE0/0/0/0
  next-hop ipv6 10:10:10::2
  delay-measurement
```

This example shows how to enable PM for link delay over an interface with only IPv6 link-local address:

```
interface TenGigE0/0/0/0
 ipv6 enable

performance-measurement
 interface TenGigE0/0/0/0
  delay-measurement
```

## Verification

```
RP/0/0/CPU0:router# show performance-measurement profile interface
Thu Dec 12 14:13:16.029 PST

------------------------------------------------------------------------------
0/0/CPU0
------------------------------------------------------------------------------
Interface Delay-Measurement:
  Profile configuration:
    Measurement Type                          : Two-Way
    Probe computation interval                : 30 (effective: 30) seconds
    Type of services                          : Traffic Class: 6, DSCP: 48
    Burst interval                            : 3000 (effective: 3000) mSec
    Burst count                               : 10 packets
    Encap mode                                : UDP
    Payload Type                              : TWAMP-light
    Destination sweeping mode                 : Disabled
    Periodic advertisement                    : Enabled
      Interval                                : 120 (effective: 120) sec
      Threshold                               : 10%
      Minimum-Change                          : 500 uSec
    Advertisement accelerated                 : Disabled
    Threshold crossing check                  : Minimum-delay

RP/0/0/CPU0:router# show performance-measurement summary detail location 0/2/CPU0

Thu Dec 12 14:09:59.162 PST

------------------------------------------------------------------------------
0/2/CPU0
------------------------------------------------------------------------------
Total interfaces                              : 1
Total SR Policies                             : 0
Total RSVP-TE tunnels                         : 0
Total Maximum PPS                             : 2000 pkts/sec
Total Interfaces PPS                          : 0 pkts/sec
Maximum Allowed Multi-hop PPS                 : 2000 pkts/sec
Multi Hop Requested PPS                       : 0 pkts/sec (0% of max allowed)
Dampened Multi Hop Requested PPS              : 0% of max allowed
Inuse Burst Interval Adjustment Factor        : 100% of configuration


Interface Delay-Measurement:
  Total active sessions                       : 1
  Counters:
    Packets:
      Total sent                              : 26
      Total received                          : 26
    Errors:
        TX:
          Reason interface down               : 0
          Reason no MPLS caps                 : 0
          Reason no IP address                : 0
          Reason other                        : 0
        RX:
          Reason negative delay               : 0
          Reason delay threshold exceeded     : 0
          Reason missing TX timestamp         : 0
          Reason missing RX timestamp         : 0
          Reason probe full                   : 0
          Reason probe not started            : 0
          Reason control code error           : 0
          Reason control code notif           : 0
```

```
    Probes:
      Total started                          : 3
      Total completed                        : 2
      Total incomplete                       : 0
      Total advertisements                   : 0

SR Policy Delay-Measurement:
  Total active sessions                      : 0
  Counters:
    Packets:
      Total sent                             : 0
      Total received                         : 0
    Errors:
        TX:
          Reason interface down              : 0
          Reason no MPLS caps                : 0
          Reason no IP address               : 0
          Reason other                       : 0
        RX:
          Reason negative delay              : 0
          Reason delay threshold exceeded    : 0
          Reason missing TX timestamp        : 0
          Reason missing RX timestamp        : 0
          Reason probe full                  : 0
          Reason probe not started           : 0
          Reason control code error          : 0
          Reason control code notif          : 0
    Probes:
      Total started                          : 0
      Total completed                        : 0
      Total incomplete                       : 0
      Total advertisements                   : 0

RSVP-TE Delay-Measurement:
  Total active sessions                      : 0
  Counters:
    Packets:
      Total sent                             : 0
      Total received                         : 0
    Errors:
        TX:
          Reason interface down              : 0
          Reason no MPLS caps                : 0
          Reason no IP address               : 0
          Reason other                       : 0
        RX:
          Reason negative delay              : 0
          Reason delay threshold exceeded    : 0
          Reason missing TX timestamp        : 0
          Reason missing RX timestamp        : 0
          Reason probe full                  : 0
          Reason probe not started           : 0
          Reason control code error          : 0
          Reason control code notif          : 0
    Probes:
      Total started                          : 0
      Total completed                        : 0
      Total incomplete                       : 0
      Total advertisements                   : 0

Global Delay Counters:
  Total packets sent                         : 26
  Total query packets received               : 26
```

```
  Total invalid session id                    : 0
  Total missing session                       : 0

RP/0/0/CPU0:router# show performance-measurement interfaces detail
Thu Dec 12 14:16:09.692 PST

--------------------------------------------------------------------------------
0/0/CPU0
--------------------------------------------------------------------------------

--------------------------------------------------------------------------------
0/2/CPU0
--------------------------------------------------------------------------------

Interface Name: GigabitEthernet0/2/0/0 (ifh: 0x1004060)
  Delay-Measurement                 : Enabled
  Loss-Measurement                  : Disabled
  Configured IPv4 Address           : 10.10.10.2
  Configured IPv6 Address           : 10:10:10::2
  Link Local IPv6 Address           : fe80::3a:6fff:fec9:cd6b
  Configured Next-hop Address       : Unknown
  Local MAC Address                 : 023a.6fc9.cd6b
  Next-hop MAC Address              : 0291.e460.6707
  Primary VLAN Tag                  : None
  Secondary VLAN Tag                : None
  State                             : Up

  Delay Measurement session:
    Session ID          : 1

    Last advertisement:
      Advertised at: Dec 12 2019 14:10:43.138 (326.782 seconds ago)
      Advertised reason: First advertisement
      Advertised delays (uSec): avg: 839, min: 587, max: 8209, variance: 297

    Next advertisement:
      Threshold check scheduled in 1 more probe (roughly every 120 seconds)
      Aggregated delays (uSec): avg: 751, min: 589, max: 905, variance: 112
      Rolling average (uSec): 756

    Current Probe:
      Started at Dec 12 2019 14:15:43.154 (26.766 seconds ago)
      Packets  9, received: 9
      Measured delays (uSec): avg: 795, min: 631, max: 1199, variance: 164
      Next probe scheduled at Dec 12 2019 14:16:13.132 (in 3.212 seconds)
      Next burst packet will be sent in 0.212 seconds
      Burst packet sent every 3.0 seconds
      Probe samples:
        Packet Rx Timestamp       Measured Delay (nsec)
        Dec 12 2019 14:15:43.156          689223
        Dec 12 2019 14:15:46.156          876561
        Dec 12 2019 14:15:49.156          913548
        Dec 12 2019 14:15:52.157         1199620
        Dec 12 2019 14:15:55.156          794008
        Dec 12 2019 14:15:58.156          631437
        Dec 12 2019 14:16:01.157          656440
        Dec 12 2019 14:16:04.157          658267
        Dec 12 2019 14:16:07.157          736880
```

You can also use the following commands for verifying the PM for link delay on the local-end router.

| Command | Description |
|---|---|
| **show performance-measurement history probe interfaces** [*interface*] | Displays the PM link-delay probe history for interfaces. |
| **show performance-measurement history aggregated interfaces** [*interface*] | Displays the PM link-delay aggregated history for interfaces. |
| **show performance-measurement history advertisement interfaces** [*interface*] | Displays the PM link-delay advertisement history for interfaces. |
| **show performance-measurement counters** [interface *interface*] [location *location-name*] | Displays the PM link-delay session counters. |

You can also use the following commands for verifying the PM for link-delay configuration on the remote-end router.

| Command | Description |
|---|---|
| **show performance-measurement responder summary** [**location** *location-name*] | Displays the PM for link-delay summary on the remote-end router (responder). |
| **show performance-measurement responder interfaces** [*interface*] | Displays PM for link-delay for interfaces on the remote-end router. |
| **show performance-measurement responder counters** [**interface** *interface*] [**location** *location-name*] | Displays the PM link-delay session counters on the remote-end router. |

### Configure a Static Delay Value on an Interface

You can configure an interface to advertise a static delay value, instead of the measured delay value. When you configure a static delay value, the advertisement is triggered immediately. The average, minimum, and maximum advertised values will use the static delay value, with a variance of 0.

Scheduled probes will continue, and measured delay metrics will be aggregated and stored in history buffer. However, advertisement threshold checks are suppressed so that there are no advertisements of the actual measured delay values. If the configured static delay value is removed, the next scheduled advertisement threshold check will update the advertised measured delay values.

The static delay value can be configured from 1 to 16777215 microseconds (16.7 seconds).

This example shows how to configure a static delay of 1000 microseconds:

```
RP/0/0/CPU0:router(config)# performance-measurement
RP/0/0/CPU0:router(config-perf-meas)# interface TenGigE0/0/0/0
RP/0/0/CPU0:router(config-pm-intf)# delay-measurement
RP/0/0/CPU0:router(config-pm-intf-dm)# advertise-delay 1000
```

### Running Configuration

```
performance-measurement
 interface GigabitEthernet0/0/0/0
  delay-measurement
   advertise-delay 1000
```

```
  !
 !
!
```

### Verification

```
RP/0/RSP0/CPU0:ios# show performance-measurement interfaces detail

-------------------------------------------------------------------------------
0/0/CPU0
-------------------------------------------------------------------------------

Interface Name: GigabitEthernet0/0/0/0 (ifh: 0x0)
  Delay-Measurement                 : Enabled

. . .

    Last advertisement:
      Advertised at: Nov 29 2021 21:53:00.656 (7.940 seconds ago)
      Advertised reason: Advertise delay config
      Advertised delays (uSec): avg: 1000, min: 1000, max: 1000, variance: 0

. . .
```

## SR Performance Measurement Named Profiles

You can create a named performance measurement profile for delay or liveness.

### Delay Profile

This example shows how to create a named SR performance measurement delay profile.

```
Router(config)# performance-measurement delay-profile sr-policy profile2
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# burst-interval 60
Router(config-pm-dm-srpolicy-probe)# computation-interval 60
Router(config-pm-dm-srpolicy-probe)# protocol twamp-light
Router(config-pm-dm-srpolicy-probe)# tos dscp 63

Router(config-pm-dm-srpolicy)# advertisement
Router(config-pm-dm-srpolicy-adv)# periodic
Router(config-pm-dm-srpolicy-adv-per)# interval 60
Router(config-pm-dm-srpolicy-adv-per)# minimum-change 1000
Router(config-pm-dm-srpolicy-adv-per)# threshold 20
Router(config-pm-dm-srpolicy-adv-per)# commit
```

Apply the delay profile for an SR Policy.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy TEST
Router(config-sr-te-policy)# color 4 end-point ipv4 10.10.10.10
Router(config-sr-te-policy)# performance-measurement
Router(config-sr-te-policy-perf-meas)# delay-measurement delay-profile name profile2

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 LIST1
Router(config-sr-te-pp-info)# weight 2

Router(config-sr-te-policy-path-pref)# explicit segment-list LIST2
Router(config-sr-te-pp-info)# weight 3
```

### Running Configuration

```
Router# show run segment-routing traffic-eng policy TEST

segment-routing
 traffic-eng
  policy TEST
   color 4 end-point ipv4 10.10.10.10
   candidate-paths
    preference 100
     explicit segment-list LIST1
      weight 2
     !
     explicit segment-list LIST2
      weight 3
     !
    !
   !
   performance-measurement
    delay-measurement
     delay-profile name profile2
```

### Verification

```
Router# show performance-measurement profile named-profile delay sr-policy name profile2

------------------------------
0/RSP0/CPU0
------------------------------
SR Policy Delay Measurement Profile Name: profile2
  Profile configuration:
    Measurement mode                              : One-way
    Protocol type                                 : TWAMP-light
    Encap mode                                    : UDP
    Type of service:
      PM-MPLS traffic class                       : 6
      TWAMP-light DSCP                            : 63
    Probe computation interval                    : 60 (effective: 60) seconds
    Burst interval                                : 60 (effective: 60) mSec
    Packets per computation interval              : 1000
    Periodic advertisement                        : Enabled
      Interval                                    : 60 (effective: 60) sec
      Threshold                                   : 20%
      Minimum-change                              : 1000 uSec
    Advertisement accelerated                     : Disabled
    Advertisement logging:
      Delay exceeded                              : Disabled (default)
    Threshold crossing check                      : Maximum-delay
    Router alert                                  : Disabled (default)
    Destination sweeping mode                     : Disabled
    Liveness detection parameters:
      Multiplier                                  : 3
      Logging state change                        : Disabled
```

### On-Demand SR Policy

```
Router(config-sr-te)# on-demand color 20
Router(config-sr-te-color)# performance-measurement delay-measurement
Router(config-sr-te-color-delay-meas)# delay-profile name profile2
Router(config-sr-te-color-delay-meas)# commit
```

### Running Configuration

```
Router# show run segment-routing traffic-eng on-demand color 20

segment-routing
 traffic-eng
  on-demand color 20
```

```
  performance-measurement
   delay-measurement
    delay-profile name profile2
```

### Liveness Profile

This example shows how to create a *named* SR performance measurement liveness profile.

```
Router(config)# performance-measurement liveness-profile sr-policy name profile3
Router(config-pm-ld-srpolicy)# probe
Router(config-pm-ld-srpolicy-probe)# burst-interval 60
Router(config-pm-ld-srpolicy-probe)# measurement-mode loopback
Router(config-pm-ld-srpolicy-probe)# tos dscp 10
Router(config-pm-ld-srpolicy-probe)# liveness-detection
Router(config-pm-ld-srpolicy-probe)# multiplier 5
Router(config-pm-ld-srpolicy-probe)# commit
```

### Apply the Liveness Profile for the SR Policy

This example shows how to enable PM for SR policy liveness for a specific policy.

For the same policy, you cannot enable delay-measurement (delay-profile) and liveness-detection (liveness-profile) at the same time. For example, if delay measurement is enabled, use the **no delay-measurement** command to disable it, and then enable the following command for enabling liveness detection.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy TRST2
Router(config-sr-te-policy)# color 40 end-point ipv4 20.20.20.20
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 50
Router(config-sr-te-policy-path-pref)# explicit segment-list LIST3
Router(config-sr-te-pp-info)# weight 2

Router(config-sr-te-policy-path-pref)# explicit segment-list LIST4
Router(config-sr-te-pp-info)# weight 3

Router(config-sr-te-policy)# performance-measurement
Router(config-sr-te-policy-perf-meas)# liveness-detection liveness-profile name profile3
```

### Running Configuration

```
Router# show run segment-routing traffic-eng policy TRST2

segment-routing
 traffic-eng
  policy TRST2
   color 40 end-point ipv4 20.20.20.20
   candidate-paths
    preference 50
     explicit segment-list LIST3
      weight 2
     !
     explicit segment-list LIST4
      weight 3
     !
    !
   !
   performance-measurement
    liveness-detection
     liveness-profile name profile3
    !
```

### Verification

```
Router# show performance-measurement profile named-profile delay

-----------
0/RSP0/CPU0
-----------

SR Policy Liveness Detection Profile Name: profile1
  Profile configuration:
    Measurement mode                          : Loopback
    Protocol type                             : TWAMP-light
    Type of service:
      TWAMP-light DSCP                        : 10
    Burst interval                            : 60 (effective: 60) mSec
    Destination sweeping mode                 : Disabled
    Liveness detection parameters:
      Multiplier                              : 3
      Logging state change                    : Disabled

SR Policy Liveness Detection Profile Name: profile3
  Profile configuration:
    Measurement mode                          : Loopback
    Protocol type                             : TWAMP-light
    Type of service:
      TWAMP-light DSCP                        : 10
    Burst interval                            : 60 (effective: 60) mSec
    Destination sweeping mode                 : Disabled
    Liveness detection parameters:
      Multiplier                              : 3
      Logging state change                    : Disabled
```

### On-Demand SR Policy

For the same policy, you cannot enable delay-measurement (delay-profile) and liveness-detection (liveness-profile) at the same time. For example, to disable delay measurement, use the **no delay-measurement** command, and then enable the following command for enabling liveness detection.

```
Router(config-sr-te)# on-demand color 30
Router(config-sr-te-color)# performance-measurement
Router(config-sr-te-color-pm)# liveness-detection liveness-profile name profile1
Router(config-sr-te-color-delay-meas)# commit
```

### Running Configuration

```
Router# show run segment-routing traffic-eng on-demand color 30

segment-routing
 traffic-eng
  on-demand color 30
   performance-measurement
    liveness-detection
     liveness-profile name profile1
    !
```

### Verification

```
Router# show performance-measurement profile named-profile liveness sr-policy name profile1

--------------------
0/RSP0/CPU0
--------------------
SR Policy Liveness Detection Profile Name: profile1
  Profile configuration:
    Measurement mode                          : Loopback
    Protocol type                             : TWAMP-light
    Type of service:
```

```
  TWAMP-light DSCP                        : 10
Burst interval                            : 60 (effective: 60) mSec
Destination sweeping mode                 : Disabled
Liveness detection parameters:
  Multiplier                              : 3
  Logging state change                    : Disabled
```

# Delay Normalization

*Table 39: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR-TE Delay Normalization for OSPF | Release 7.3.1 | This feature extends the current Delay Normalization feature to support OSPF. |

Performance measurement (PM) measures various link characteristics like packet loss and delay. Such characteristics can be used by IS-IS as a metric for Flexible Algorithm computation. Low latency routing using dynamic delay measurement is one of the primary use cases for Flexible Algorithm technology.

Delay is measured in microseconds. If delay values are taken as measured and used as link metrics during the IS-IS topology computation, some valid ECMP paths might be unused because of the negligible difference in the link delay.

The Delay Normalization feature computes a normalized delay value and uses the normalized value instead. This value is advertised and used as a metric during the Flexible Algorithm computation.

The normalization is performed when the delay is received from the delay measurement component. When the next value is received, it is normalized and compared to the previous saved normalized value. If the values are different, then the LSP generation is triggered.

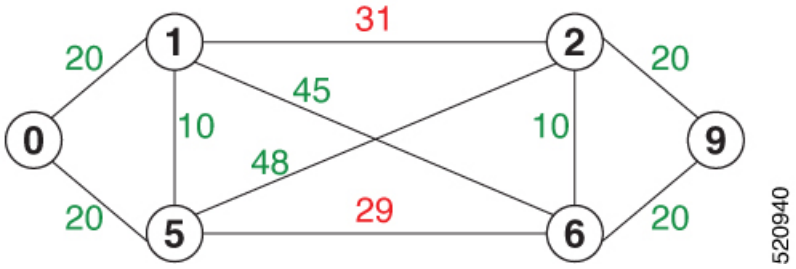
The following formula is used to calculate the normalized value:

- **Dm** – measured Delay
- **Int** – configured normalized Interval
- **Off** – configured normalized Offset (must be less than the normalized interval Int)
- **Dn** – normalized Delay
- **a** = Dm / Int (rounded down)
- **b** = a * Int + Off

If the measured delay (Dm) is less than or equal to **b**, then the normalized delay (Dn) is equal to **b**. Otherwise, Dn is **b** + **Int**.

### Example

The following example shows a low-latency service. The intent is to avoid high-latency links (1-6, 5-2). Links 1-2 and 5-6 are both low-latency links. The measured latency is not equal, but the difference is insignificant.




We can normalize the measured latency before it is advertised and used by IS-IS. Consider a scenario with the following:

- Interval = 10
- Offset = 3

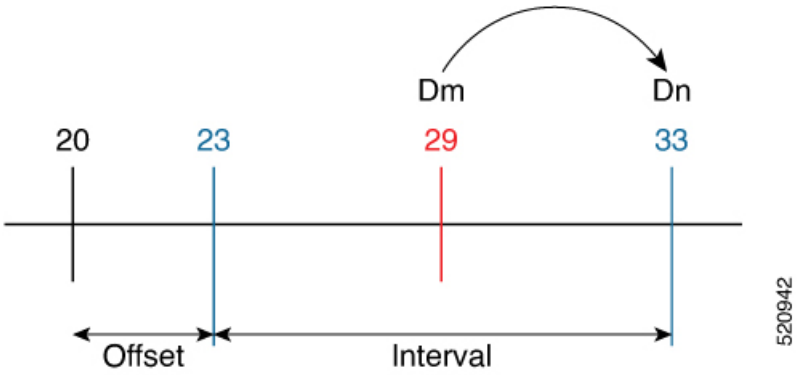
The measured delays will be normalized as follows:

- **Dm** = 29

  **a** = 29 / 10 = 2 (2.9, rounded down to 2)

  **b** = 2 * 10 + 3 = 23

  In this case, **Dm** (29) is greater than **b** (23); so **Dn** is equal to **b**+**I** (23 + 10) = **33**

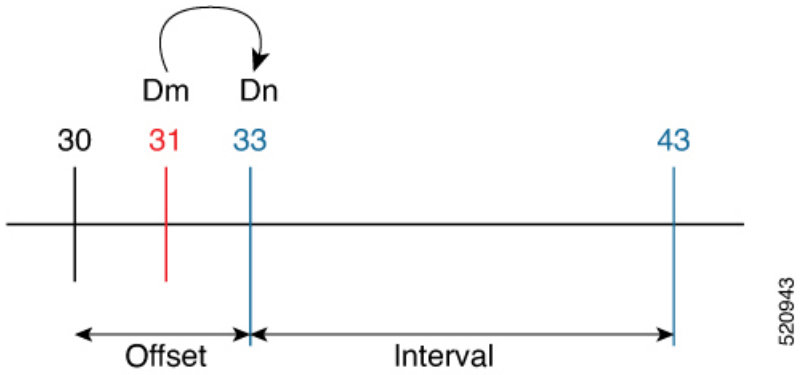



- **Dm** = 31

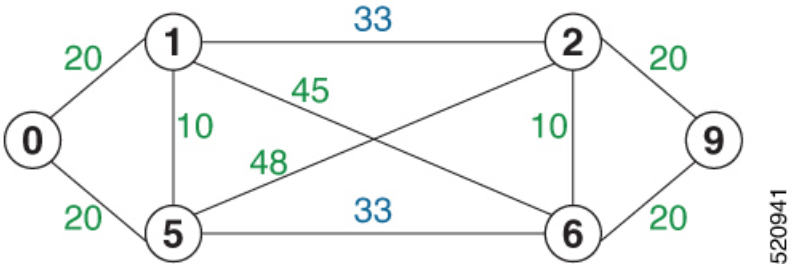
  **a** = 31 / 10 = 3 (3.1, rounded down to 3)

  **b** = 3 * 10 + 3 = 33

  In this case, **Dm** (31) is less than **b** (33); so **Dn** is **b** = **33**




The link delay between 1-2 and 5-6 is normalized to 33.




### Configuration

Delay normalization is disabled by default. To enable and configure delay normalization, use the **delay normalize interval** *interval* [**offset** *offset*] command.

- *interval* – The value of the normalize interval in microseconds.
- *offset* – The value of the normalized offset in microseconds. This value must be smaller than the value of normalized interval.

### IS-IS Configuration

```
router isis 1
 interface GigEth 0/0/0/0
  delay normalize interval 10 offset 3
  address-family ipv4 unicast
   metric 77
```

### OSPF Configuration

```
router ospf 1
 area 0
  interface GigabitEthernet0/0/0/0
   delay normalize interval 10 offset 3
  !
 !
!
```

# Link Anomaly Detection with IGP Penalty

*Table 40: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Link Anomaly Detection with IGP Penalty | Release 7.4.1 | This feature allows you to define thresholds above the measured delay that is considered "anomalous" or unusual. When this threshold is exceeded, an anomaly (A) bit/flag is set along with link delay attribute that is sent to clients. |

Customers might experience performance degradation issues, such as increased latency or packet loss on a link. Degraded links might be difficult to troubleshoot and can affect applications, especially in cases where traffic is sent over multiple ECMP paths where one of those paths is degraded.

The Anomaly Detection feature allows you to define a delay anomaly threshold to identify unacceptable link delays. Nodes monitor link performance using link delay monitoring probes. The measured value is compared against the delay anomaly threshold values. When the upper bound threshold is exceeded, the link is declared "abnormal", and performance measurement sets an anomaly bit (A-bit). When IGP receives the A-bit, IGP can automatically increase the IGP metric of the link by a user-defined amount to make this link undesirable or unusable. When the link recovers (lower bound threshold), PM resets the A-bit.

For information on configuring IGP penality, see the following:

- IS-IS Penalty for Link Delay Anomaly
- OSPF Penalty for Link Delay Anomaly

### Usage Guidelines and Limitations

This feature is not active when narrow metrics are configured because the performance measurement advertisement requires the "wide" metric type length values.

### Configuration Example

The following example shows how to configure the upper and lower anomoly thresholds. The range for *upper_bound* and *lower_bound* is from 1 to 200,000 microseconds. The *lower_bound* value must be less than the *upper_bound* value.

```
RP/0/0/CPU0:router(config)# performance-measurement delay-profile interfaces default
RP/0/0/CPU0:router(config-pm-dm-intf)# advertisement
RP/0/0/CPU0:router(config-pm-dm-intf-adv)# anomaly-check upper-bound 5000 lower-bound 1000
RP/0/0/CPU0:router(config-pm-dm-intf-adv)# commit
```

### Running Configuration

```
performance-measurement
 delay-profile interfaces default
  advertisement
   anomaly-check
    upper-bound 5000 lower-bound 1000
   !
  !
```

```
 !
!
end
```

# Delay Measurement for IP Endpoint

*Table 41: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| IP Endpoint Delay Measurement Monitoring | Release 7.4.1 | This feature measures the end-to-end delay and monitors liveness of a specified IP endpoint node, including VRF-aware (awareness of multiple customers belonging to different VRFs).<br>This feature is supported on IPv4, IPv6, and MPLS data planes. |

Delay for an IP endpoint is the amount of time it takes for a data packet to travel from a source device to a specific IP endpoint within a network.

To measure a delay for a packet, also called a probe, is sent from a source device to the target IP endpoint.

The time from when the packet leaves the source to when it arrives at the endpoint is measured and recorded as the delay.

You can measure one-way delay, Two-way delay, and Roundtrip delay or delay in loop-back mode. For more information on Delay measurement, see Link Delay Measurement and Measurement Modes.

### Collecting IP Endpoint Probe Statistics

- Statistics associated with the probe for delay metrics are available via Histogram and Streaming Telemetry.
- Model Driven Telemetry (MDT) is supported for the following data:
  - Summary, endpoint, session, and counter show command bags.
  - History buffers data
- Model Driven Telemetry (MDT) and Event Driven Telemetry (EDT) are supported for the following data:
  - Delay metrics computed in the last probe computation-interval (event: probe-completed)
  - Delay metrics computed in the last aggregation-interval; that is, end of the periodic advertisement-interval (event: advertisement-interval expired)
  - Delay metrics last notified (event: notification-triggered)
- The following xpaths for MDT/EDT is supported:
  - `Cisco-IOS-XR-perf-meas-oper:performance-measurement/nodes/node/endpoints/endpoint-delay/endpoint-last-probes`
  - `Cisco-IOS-XR-perf-meas-oper:performance-measurement/nodes/node/endpoints/endpoint-delay/endpoint-last-aggregations`

- `Cisco-IOS-XR-perf-meas-oper:performance-measurement/nodes/node/endpoints/endpoint-delay/endpoint-last-advertisements`

### Guidelines and Limitations

You can specify a custom labeled path through one or more user-configured segment-lists. User-configured segment-list represents the forwarding path from sender to reflector when the probe is configured in delay-measurement mode.

- Examples of the custom segment-list include:
  - Probe in delay-measurement mode with a segment-list that includes Flex-Algo prefix SID of the endpoint
  - Probe in delay-measurement mode with a segment-list that includes a SID-list with labels to reach the endpoint or the sender (forward direction)
  - Probe in delay-measurement mode with a segment-list that includes BSID associated with SR policy to reach the end point.
- Endpoint segment list configuration is not supported under nondefault VRF.
- SR Performance Measurement endpoint session over BVI interface is not supported.

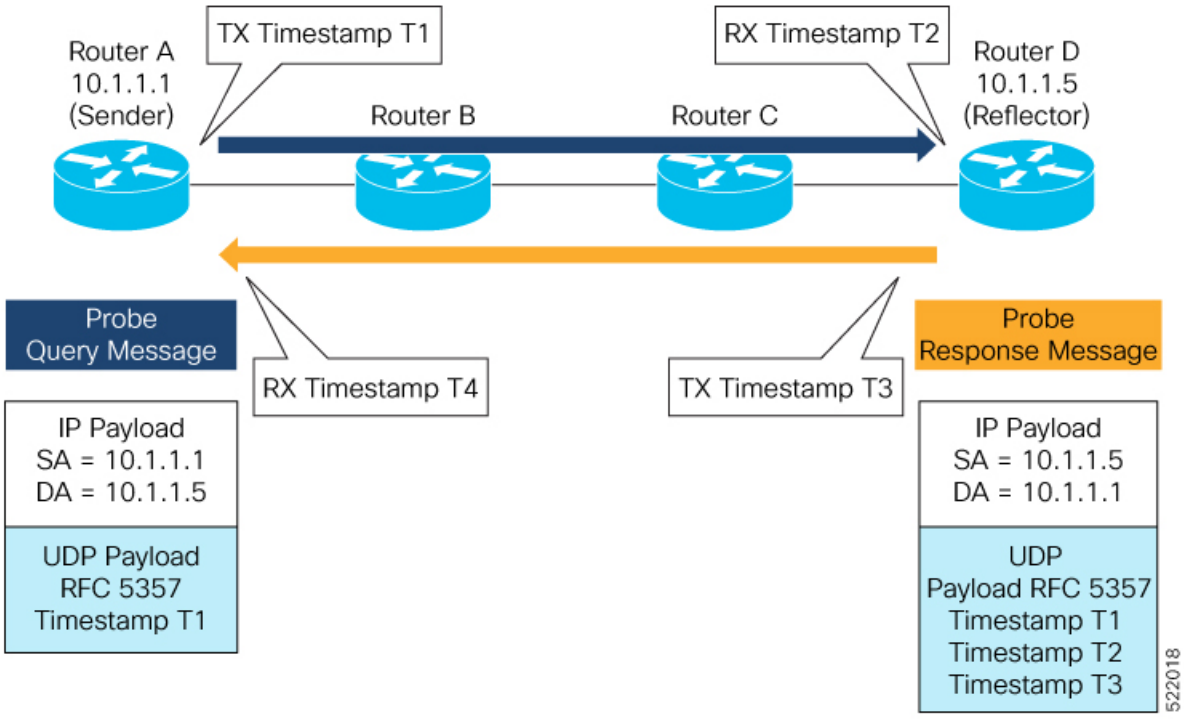
## IP Endpoint Delay Measurement over MPLS Network Usecases

The following use-cases show different ways to deploy delay measurement and liveness detection for IP endpoints.

### Use-Case 1: Delay Measurement Probe Toward an IP Endpoint Reachable in the Global Routing Table

The following figure illustrates a delay measurement probe toward an IP endpoint reachable in the global routing table. The network interconnecting the sender and the reflector provides plain IP connectivity.

*Figure 26: Delay Measurement Probe Toward an IP Endpoint Reachable in the Global Routing Table*




## Configuration

```
RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 10.1.1.5
RouterA(config-pm-ep)# source-address ipv4 10.1.1.1
RouterA(config-pm-ep)# delay-measurement
RouterA(config-pm-ep-dm)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# delay-profile endpoint default
RouterA(config-pm-dm-ep)# probe
RouterA(config-pm-dm-ep-probe)# measurement-mode one-way
```

## Running Configuration

```
performance-measurement
 endpoint ipv4 10.1.1.5
  source-address ipv4 10.1.1.1
  delay-measurement
  !
 !
 delay-profile endpoint default
  probe
   measurement-mode one-way
  !
 !
!
```

## Verification

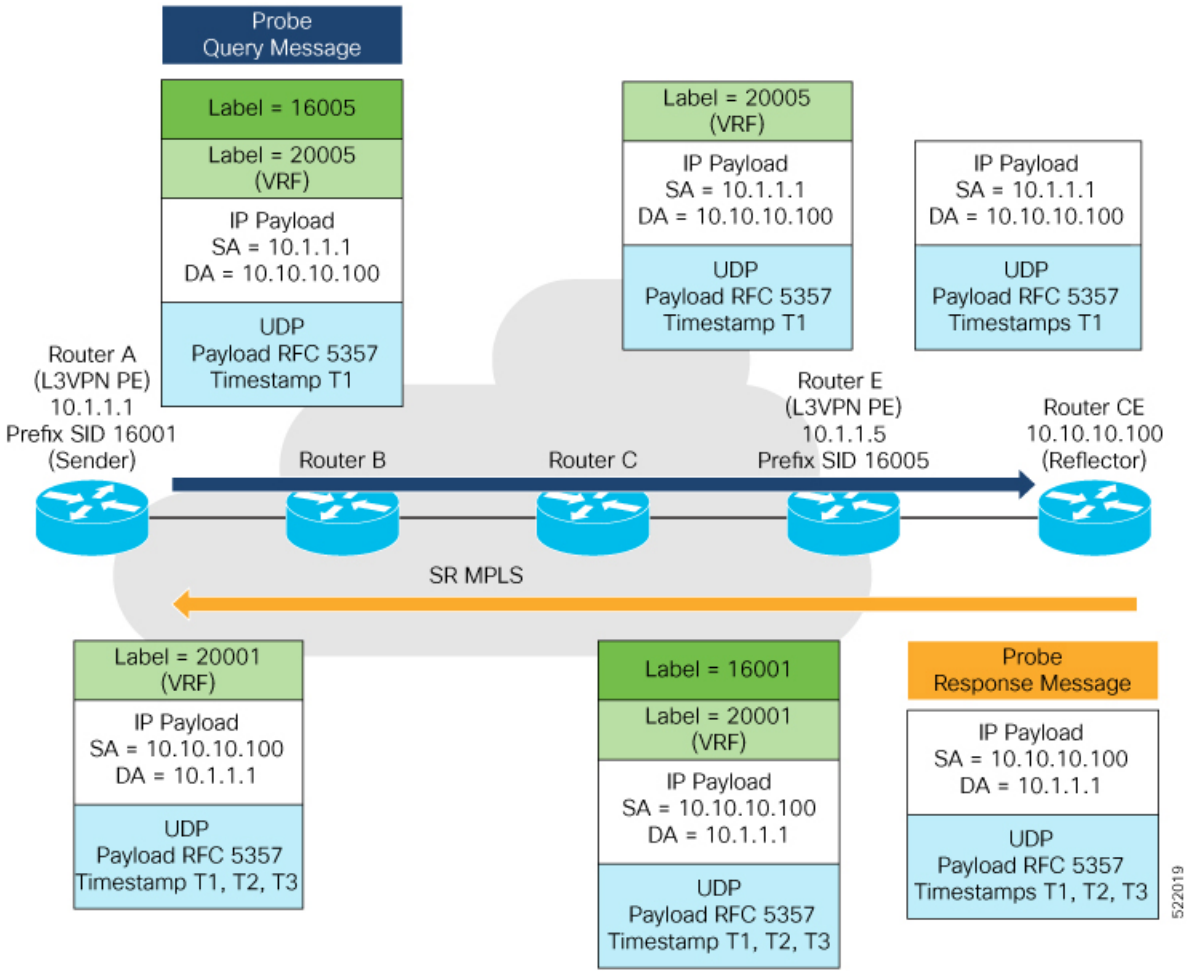
```
RouterA# show performance-measurement endpoint ipv4 10.1.1.5
```

```
0/RSP0/CPU0
------------------------------------------------------------------------------------------------------------------------

Endpoint name: IPv4-10.1.1.5-vrf-default
  Source address               : 10.1.1.1
  VRF name                     : default
  Delay-measurement            : Enabled
  Description                  : Not set
  Profile Keys:
    Profile name               : default
    Profile type               : Endpoint Delay Measurement


  Segment-list                 : None
  Delay Measurement session:
    Session ID          : 33554433
    Last advertisement:
      No advertisements have occured

    Next advertisement:
      Threshold check scheduled in 4 more probes (roughly every 120 seconds)
      No probes completed

    Current computation:
      Started at: Jul 19 2021 16:28:06.723 (17.788 seconds ago)
      Packets  6, received: 0
      Measured delays (uSec): avg: 0, min: 0, max: 0, variance: 0
      Next probe scheduled at: Jul 19 2021 16:28:36.718 (in 12.207 seconds)
      Next burst packet will be sent in 0.207 seconds
      Burst packet sent every 3.0 seconds
```

### Use-Case 2: Delay Measurement Probe Toward an IP Endpoint Reachable in a User-Specified VRF

The following figure illustrates a delay measurement probe toward an IP endpoint reachable in a user-specified L3VPN's VRF routing table. The L3VPN ingress PE (Router A) acts as the sender. The reflector is located in a CE device behind the L3VPN egress PE (Router E). The network interconnecting the L3VPN PEs provides MPLS connectivity with Segment Routing.

*Figure 27: Delay Measurement Probe Toward an IP Endpoint Reachable in a User-Specified VRF*




## Configuration

```
RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 10.10.10.100 vrf green
RouterA(config-pm-ep)# source-address ipv4 10.1.1.1
RouterA(config-pm-ep)# delay-measurement
RouterA(config-pm-ep-dm)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# delay-profile endpoint default
RouterA(config-pm-dm-ep)# probe
RouterA(config-pm-dm-ep-probe)# measurement-mode one-way
```

## Running Configuration

```
performance-measurement
 endpoint ipv4 10.10.10.100 vrf green
  source-address ipv4 10.1.1.1
  delay-measurement
  !
 !
 delay-profile endpoint default
  probe
   measurement-mode one-way
```

```
  !
 !
!
```

### Verification

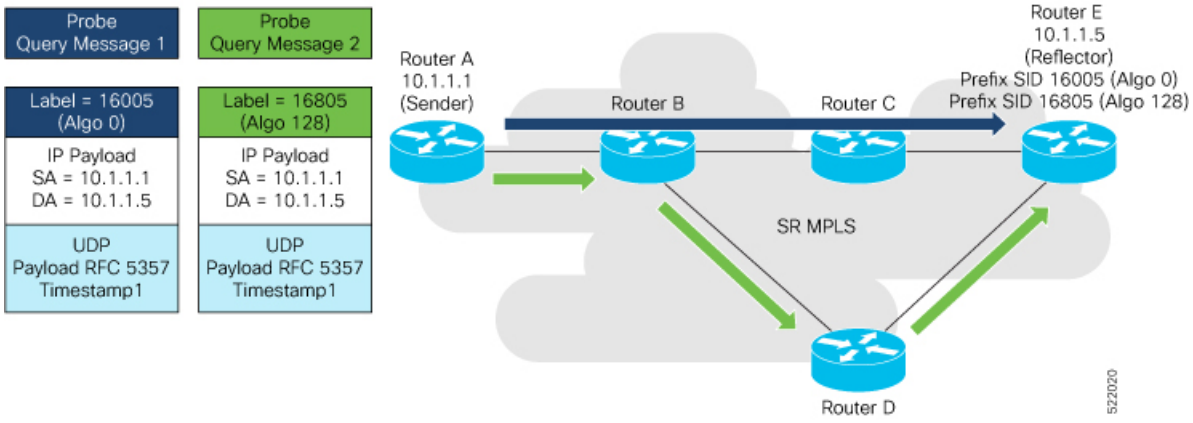
```
RouterA# show performance-measurement endpoint vrf green

----------------------------------------------------------------------------------------------------------------------------

0/RSP0/CPU0

----------------------------------------------------------------------------------------------------------------------------

Endpoint name: IPv4-10.10.10.100-vrf-green
  Source address               : 10.1.1.1
  VRF name                     : green
  Delay-measurement            : Enabled
  Description                  : Not set
  Profile Keys:
    Profile name               : default
    Profile type               : Endpoint Delay Measurement


  Segment-list                 : None
  Delay Measurement session:
    Session ID          : 33554434
    Last advertisement:
      No advertisements have occured

    Next advertisement:
      Advertisement not scheduled as the probe is not running

    Current computation:
      Not running: Unable to resolve (non-existing) vrf
```

### Use Case 3: Delay Measurement Probe Toward an IP Endpoint Using Custom Labeled Paths

The following figure illustrates a delay measurement probe toward an IP endpoint learned by the IGP. The network interconnecting the sender and reflector provides MPLS connectivity with Segment Routing.

The IP endpoint is advertised with multiple SR algorithms (Algo 0 and Flex Algo 128). The probe is configured with two custom-labeled paths in order to monitor the LSP for each algorithm separately.

***Figure 28: Delay Measurement Probe Toward an IP Endpoint Using Custom Labeled Paths***



**Note** The probe response messages are not shown in the above figure.

### Configuration

```
RouterA(config)# segment-routing
RouterA(config-sr)# traffic-eng
RouterA(config-sr-te)# segment-list name SIDLIST1-Algo0
RouterA(config-sr-te-sl)# index 10 mpls label 16005
RouterA(config-sr-te-sl)# exit

RouterA(config-sr-te)# segment-list name SIDLIST2-FlexAlgo128
RouterA(config-sr-te-sl)# index 10 mpls label 16085
RouterA(config-sr-te-sl)# exit
RouterA(config-sr-te)# exit
RouterA(config-sr)# exit

RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 10.1.1.5
RouterA(config-pm-ep)# source-address ipv4 10.1.1.1
RouterA(config-pm-ep)# segment-list name SIDLIST1-Algo0
RouterA(config-pm-ep-sl)# exit
RouterA(config-pm-ep)# segment-list name SIDLIST2-FlexAlgo128
RouterA(config-pm-ep-sl)# exit
RouterA(config-pm-ep)# delay-measurement
RouterA(config-pm-ep-dm)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# delay-profile endpoint default
RouterA(config-pm-dm-ep)# probe
RouterA(config-pm-dm-ep-probe)# measurement-mode one-way
```

### Running Configuration

```
segment-routing
 traffic-eng
  segment-list SIDLIST1-Algo0
   index 10 mpls label 16005
   !
  segment-list SIDLIST2-FlexAlgo128
   index 10 mpls label 16085
   !
  !
 !
!
performance-measurement
 endpoint ipv4 10.1.1.5
  segment-list name SIDLIST1-Algo0
  !
  segment-list name SIDLIST2-FlexAlgo128
  !
  source-address ipv4 10.1.1.1
  delay-measurement
  !
 !
 delay-profile endpoint default
  probe
   measurement-mode one-way
  !
 !
!
```

### Verification

```
RouterA# show performance-measurement endpoint ipv4 10.1.1.5

------------------------------------------------------------------------------------------------------------------------
0/RSP0/CPU0
------------------------------------------------------------------------------------------------------------------------

Endpoint name: IPv4-10.1.1.5-vrf-default
  Source address              : 10.1.1.1
  VRF name                    : default
  Delay-measurement           : Enabled
  Description                 : Not set
  Profile Keys:
    Profile name              : default
    Profile type              : Endpoint Delay Measurement


  Segment-list                : None
  Delay Measurement session:
    Session ID          : 33554433
    Last advertisement:
      No advertisements have occured

    Next advertisement:
      Threshold check scheduled in 4 more probes (roughly every 120 seconds)
      No probes completed

    Current computation:
      Started at: Jul 19 2021 16:31:53.827 (15.844 seconds ago)
      Packets  6, received: 0
      Measured delays (uSec): avg: 0, min: 0, max: 0, variance: 0
      Next probe scheduled at: Jul 19 2021 16:32:22.957 (in 13.286 seconds)
      Next burst packet will be sent in 1.286 seconds
      Burst packet sent every 3.0 seconds

  Segment-list                : SIDLIST1-Algo0
  Delay Measurement session:
    Session ID          : 33554435
    Last advertisement:
      No advertisements have occured

    Next advertisement:
      Threshold check scheduled in 4 more probes (roughly every 120 seconds)
      No probes completed

    Current computation:
     Started at: Jul 19 2021 16:31:53.827 (15.844 seconds ago)
      Packets  4, received: 0
      Measured delays (uSec): avg: 0, min: 0, max: 0, variance: 0
      Next probe scheduled at: Jul 19 2021 16:32:22.957 (in 13.286 seconds)
      Next burst packet will be sent in 2.940 seconds
      Burst packet sent every 3.0 seconds


  Segment-list                : SIDLIST2-FlexAlgo128
  Delay Measurement session:
    Session ID          : 33554436
    Last advertisement:
      No advertisements have occured

    Next advertisement:
      Threshold check scheduled in 4 more probes (roughly every 120 seconds)
      No probes completed

    Current computation:
```

```
Started at: Jul 19 2021 16:31:53.827 (15.844 seconds ago)
Packets  4, received: 0
Measured delays (uSec): avg: 0, min: 0, max: 0, variance: 0
Next probe scheduled at: Jul 19 2021 16:32:22.957 (in 13.286 seconds)
Next burst packet will be sent in 2.940 seconds
Burst packet sent every 3.0 seconds
```

### Use-Case 4: Liveness Detection Probe Toward an IP Endpoint

IP endpoint liveness detection leverages the loopback measurement-mode. The following workflow describes the sequence of events.

1. The sender creates and transmits the PM probe packets.

   The IP destination address (DA) on the probe packets is set to the loopback value of the sender itself.

   The transmit timestamp (T1) is added to the payload.

   The probe packet is encapsulated with the label corresponding to the endpoint.

2. The network delivers the PM probe packets following the LSP toward the endpoint.

3. The end-point receives the PM probe packets.

   Packets are forwarded back to the sender based on the forwarding entry associated with the IP DA of the PM probe packet. If an LSP exists, the probe packet is encapsulated with the label of the sender.

4. The sender node receives the PM probe packets.

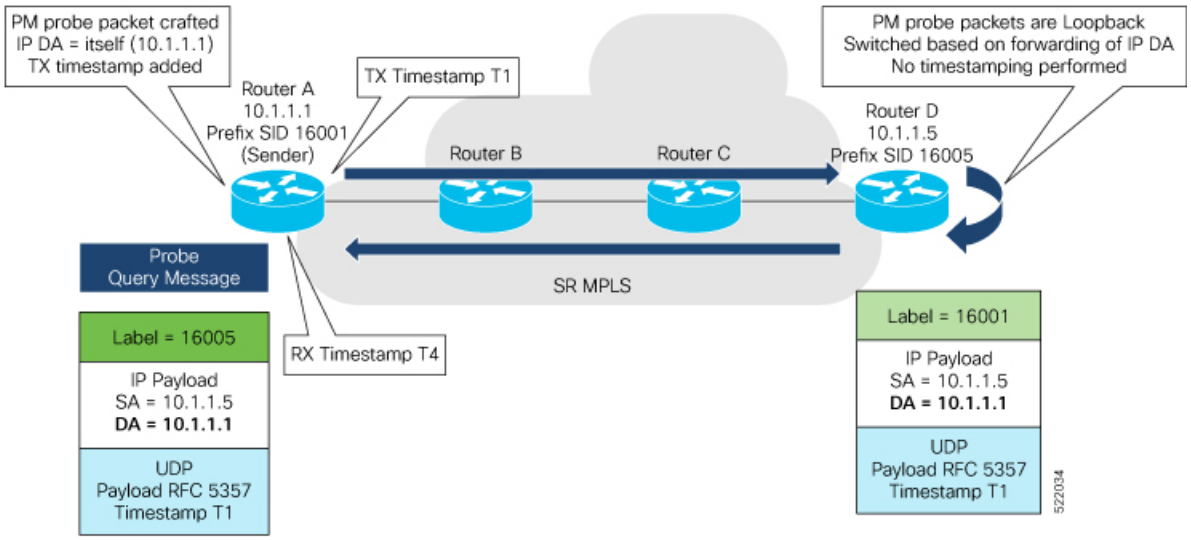
   The received timestamp (T4) stored.

   If the sender node doesn't receive the specified number of probe packets (based on the configured multiplier), the sender node declares the PM session as down.

The following figure illustrates a liveness detection probe toward an IP endpoint learned by the IGP. The network interconnecting the sender and reflector provides MPLS connectivity with Segment Routing.

The liveness detection multiplier is set to 5 to specify the number of consecutive missed probe packets before the PM session is declared as down.

***Figure 29: IP Endpoint Liveness Detection***



### Configuration

```
RouterA(config)# performance-measurement
RouterA(config-perf-meas)# endpoint ipv4 10.1.1.5
RouterA(config-pm-ep)# source-address ipv4 10.1.1.1
RouterA(config-pm-ep)# liveness-detection
RouterA(config-pm-ep-ld)# exit
RouterA(config-pm-ep)# exit
RouterA(config-perf-meas)# liveness-profile endpoint default
RouterA(config-pm-ld-ep)# liveness-detection
RouterA(config-pm-ld-ep-ld)# multiplier 5
RouterA(config-pm-ld-ep-ld)# exit
RouterA(config-pm-ld-ep)# probe
RouterA(config-pm-ld-ep-probe)# measurement-mode loopback
```

### Running Configuration

```
performance-measurement
 endpoint ipv4 10.1.1.5
  source-address ipv4 10.1.1.1
  liveness-detection
  !
 !
 liveness-profile endpoint default
  liveness-detection
   multiplier 5
  !
  probe
   measurement-mode loopback
  !
 !
!
end
```

### Verification

```
RouterA# show performance-measurement endpoint ipv4 10.1.1.5

------------------------------------------------------------------------------
0/RSP0/CPU0
------------------------------------------------------------------------------

Endpoint name: IPv4-10.1.1.5-vrf-default
  Source address              : 10.1.1.1
  VRF name                    : default
  Liveness Detection          : Enabled
  Profile Keys:
    Profile name              : default
    Profile type              : Endpoint Liveness Detection

  Segment-list                : None
  Session State: Down
  Missed count: 0
```

# SR Policy End-to-End Delay Measurement

*Table 42: Feature History Table*

| Feature Name | Release | Description |
|---|---|---|
| Segment Routing Performance Measurement for Link Delay and SR Policy Delay Using RFC 5357 (TWAMP Light) Encoding | Release 7.2.2 | This feature introduces support for Two-Way Active Measurement Protocol (TWAMP) Light (RFC 5357) for link delay and SR policy delay measurement. TWAMP Light adds two-way or round-trip measurement capabilities.<br><br>Network performance data such as packet loss, delay and delay variation, and bandwidth utilization is a critical measure for Traffic Engineering (TE). This data provides service providers the characteristics of their networks for performance evaluation that is required to ensure the Service Level Agreements (SLAs). The performance measurement and delay variation feature allows you to measure those metrics and advertise them through IGP extensions as extended TE metrics. |

The PM for SR Policy uses the IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy. In the case of TWAMP Light, the Session-Reflector doesn't necessarily know about the session state. The Session-Reflector simply copies the Sequence Number of the received packet to the Sequence Number field of the reflected packet. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

The extended TE link delay metric (minimum-delay value) can be used to compute paths for SR policies as an optimization metric or as an accumulated delay bound.

There is a need to monitor the end-to-end delay experienced by the traffic sent over an SR policy to ensure that the delay does not exceed the requested "upper-bound" and violate SLAs. You can verify the end-to-end delay values before activating the candidate-path or the segment lists of the SR policy in forwarding table, or to deactivate the active candidate-path or the segment lists of the SR policy in forwarding table.

**Note** The end-to-end delay value of an SR policy will be different than the path computation result (for example, the sum of TE link delay metrics) due to several factors, such as queuing delay within the routers.

### Usage Guidelines and Limitations for PM for SR Policy Delay

The following usage guidelines and limitations apply:

- SR-PM delay measurement over SR Policy is supported on manually configured SR Policies and On-Demand SR Policies (ODN).
- SR-PM delay measurement over SR Policy is not supported on PCE-initiated SR Policies.
- Hardware clocks must be synchronized between the querier and the responder nodes of the link using PTP for one-way delay measurement.

### Configuring Performance Measurement Parameters

This example shows how to configure performance-measurement parameters for SR policy delay as a global default profile. The default values for the different parameters in the PM for SR policy delay is given as follows:

- **probe**: The default mode for probe is one-way delay measurement. Two-way delay and loopback modes are supported. See Measurement Modes, on page 334 for more information.
- **burst interval**: Interval for sending probe packet. The default value is 3000 milliseconds and the range is from 30 to 15000 milliseconds.
- **computation interval**: Interval for metric computation. Default is 30 seconds; range is 1 to 3600 seconds.
- **protocol**:
  - **twamp-light**: SR Policy delay measurement using RFC 5357 with IP/UDP encap. This is the default protocol.
- **tos**: Type of Service
  - **dscp** *value*: The default value is 48 and the range is from 0 to 63.
  - **traffic-class** *value*: The default value is 6 and the range is from 0 to 7.
- **advertisement threshold-check**: minimum-delay/maximum-delay - The default value of periodic advertisement threshold-check is maximum-delay.
- **periodic advertisement**: Periodic advertisement is enabled by default.
- **periodic-advertisement interval**: The default value is 120 seconds and the interval range is from 30 to 3600 seconds.
- **periodic-advertisement threshold**: Checks the minimum-delay metric change for threshold crossing for periodic advertisement. The default value is 10 percent and the range is from 0 to 100 percent.
- **periodic-advertisement minimum-change**: The default value is 500 microseconds (usec) and the range is from 0 to 100000 microseconds.
- **accelerated advertisement**: Accelerated advertisement is disabled by default.
- **accelerated-advertisement threshold**: Checks the minimum-delay metric change for threshold crossing for accelerated advertisement. The default value is 20 percent and the range is from 0 to 100 percent.
- **accelerated-advertisement minimum**: The default value is 500 microseconds and the range is from 1 to 100000 microseconds.

```
Router(config)# performance-measurement delay-profile sr-policy
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# burst-interval 60
Router(config-pm-dm-srpolicy-probe)# computation-interval 60
Router(config-pm-dm-srpolicy-probe)# protocol twamp-light
Router(config-pm-dm-srpolicy-probe)# tos dscp 63
Router(config-pm-dm-srpolicy-probe)# exit

Router(config-pm-dm-srpolicy)# advertisement
Router(config-pm-dm-srpolicy-adv)# periodic
Router(config-pm-dm-srpolicy-adv-per)# interval 60
Router(config-pm-dm-srpolicy-adv-per)# minimum-change 1000
Router(config-pm-dm-srpolicy-adv-per)# threshold 20
Router(config-pm-dm-srpolicy-adv-per)# exit

Router(config-pm-dm-srpolicy-adv)# accelerated
Router(config-pm-dm-srpolicy-adv-acc)# minimum-change 1000
Router(config-pm-dm-srpolicy-adv-acc)# threshold 10
Router(config-pm-dm-srpolicy-adv-acc)# exit

Router(config-pm-dm-srpolicy-adv)# threshold-check minimum-delay
Router(config-pm-dm-srpolicy-adv)# exit
Router(config-pm-dm-srpolicy)#
```

### Configure the UDP Destination Port

Configuring the UDP port for TWAMP-Light protocol is optional. By default, PM uses port 862 as the TWAMP-reserved UDP destination port for delay.

The UDP port is configured for each PM measurement probe type (delay, loss, protocol, authentication mode, etc.) on querier and responder nodes. If you configure a different UDP port, the UDP port for each PM measurement probe type must match on the querier and the responder nodes.

**Note** The same UDP destination port is used for delay measurement for links and SR Policy.

This example shows how to configure the UDP destination port for delay.

```
Router(config)# performance-measurement

Router(config-perf-meas)# protocol twamp-light

Router(config-pm-protocol)# measurement delay unauthenticated
Router(config-pm-proto-mode)# querier-dst-port 12000
```

### Enable Performance Measurement for SR Policy

This example shows how to enable PM for SR policy delay for a specific policy.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy foo
Router(config-sr-te-policy)# performance-measurement
Router(config-sr-te-policy-perf-meas)# delay-measurement
```

### SR Policy Probe IP/UDP ECMP Hashing Configuration

This example shows how to configure SR Policy ECMP IP-hashing mode.

- The destination IPv4 address 127.x.x.x – 127.y.y.y is used in the Probe messages to take advantages of 3-tuple IP hashing (source-address, destination-address, and local router ID) for ECMP paths of SR-MPLS Policy.

**Note** The destination IPv4 address must be 127/8 range (loopback), otherwise it will be rejected.

- One PM session is always created for the actual endpoint address of the SR Policy.
- You can specify the number of IP addresses to sweep. The range is from 0 (default, no sweeping) to 128.
- Platforms may have a limitation for large label stack size to not check IP address for hashing.

```
Router(config)# performance-measurement delay-profile sr-policy
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# sweep
Router(config-pm-dm-srpolicy-probe-sweep)# destination ipv4 127.0.0.1 range 28
```

### Verification

```
Router# show performance-measurement sr-policy
Mon Jan 20 18:48:41.002 PST

------------------------------------------------------------------------------
0/0/CPU0
------------------------------------------------------------------------------

Policy Name                LSP ID Tx/Rx         Avg/Min/Max/Variance
 ---------------------------------------------------------------------------
srte_c_10_ep_192.168.0.4   2      6/6           27012/26906/27203/106

Router# show performance-measurement sr-policy name srte_c_10_ep_192.168.0.4 detail verbose
Mon Jan 20 18:44:22.400 PST

------------------------------------------------------------------------------
0/0/CPU0
------------------------------------------------------------------------------
SR Policy name: srte_c_10_ep_192.168.0.4
  Color                       : 10
  Endpoint                    : 192.168.0.4
  Number of candidate-paths   : 1

  Candidate-Path:
    Instance                  : 2
    Preference                : 100
    Protocol-origin           : Configured
    Discriminator             : 100
    Source address            : 192.168.0.2
    Reverse path label        : Not configured
    Number of segment-lists   : 1
    Last advertisement:
      No advertisements have occured
    Next advertisement:
      Check scheduled at the end of the current probe (roughly every 30 seconds)
      Aggregated delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
      Rolling average (uSec): 45218
    Last probe:
```

```
      Packets  9, received: 9
      Measured delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
    Current Probe:
      Started at Jan 20 2020 18:44:19.170 (3.453 seconds ago)
      Packets  3, received: 3      Measured delays (uSec): avg: 26588, min: 26558, max:
26630, variance: 30
    Next probe scheduled at Jan 20 2020 18:44:34.166 (in 11.543 seconds)
    Next burst packet will be sent in 1.543 seconds
    Burst packet sent every 5.0 seconds
    Liveness Detection: Disabled

    Segment-List              : R4
        16004
      Number of atomic paths  : 3
      Last advertisement:
        No advertisements have occured
      Next advertisement:
        Aggregated delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
        Rolling average (uSec): 45218
      Last probe:
        Packets  9, received: 9
        Measured delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
      Current probe:
        Packets  3, received: 3
        Measured delays (uSec): avg: 26588, min: 26558, max: 26630, variance: 30
      Liveness Detection: Disabled

      Atomic path:
        Hops                  : 127.0.0.0
        Session ID            : 33554434
        Last advertisement:
          No advertisements have occured
        Next advertisement:
          Aggregated delays (uSec): avg: 45407, min: 26629, max: 82600, variance: 18778
          Rolling average (uSec): 45407
        Last Probe:
          Packets  3, received: 3
          Measured delays (uSec): avg: 45407, min: 26629, max: 82600, variance: 18778
        Current Probe:
          Packets  1, received: 1
          Measured delays (uSec): avg: 26630, min: 26630, max: 26630, variance: 0
        Probe samples:
          Packet Rx Timestamp       Measured Delay (nsec)
          Jan 20 2020 18:44:19.198        26630730
        Liveness Detection: Disabled

      Atomic path:
        Hops                  : 127.0.0.1
        Session ID            : 33554435
        Last advertisement:
          No advertisements have occured
        Next advertisement:
          Aggregated delays (uSec): avg: 45128, min: 26521, max: 81961, variance: 18607
          Rolling average (uSec): 45128
        Last Probe:
          Packets  3, received: 3
          Measured delays (uSec): avg: 45128, min: 26521, max: 81961, variance: 18607
        Current Probe:
          Packets  1, received: 1
          Measured delays (uSec): avg: 26576, min: 26576, max: 26576, variance: 0
        Probe samples:
          Packet Rx Timestamp       Measured Delay (nsec)
          Jan 20 2020 18:44:19.198        26576938
        Liveness Detection: Disabled
```

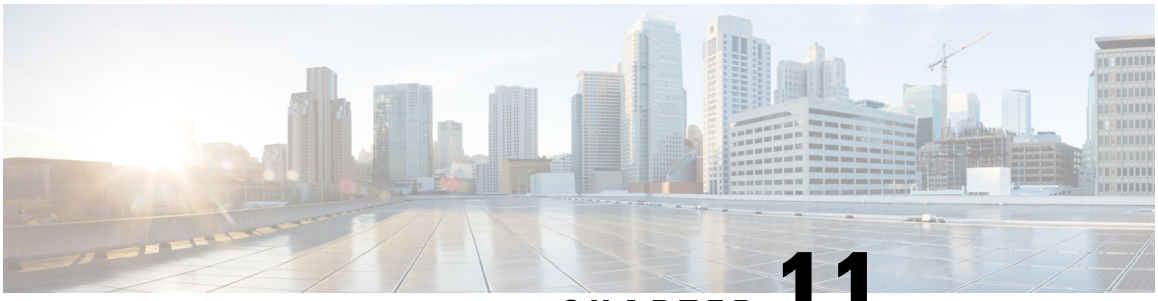
```
      Atomic path:
        Hops                  : 192.168.0.4
        Session ID            : 33554433
        Last advertisement:
          No advertisements have occured
        Next advertisement:
          Aggregated delays (uSec): avg: 45119, min: 26512, max: 81956, variance: 18607
          Rolling average (uSec): 45119
        Last Probe:
          Packets  3, received: 3
          Measured delays (uSec): avg: 45119, min: 26512, max: 81956, variance: 18607
        Current Probe:
          Packets  1, received: 1
          Measured delays (uSec): avg: 26558, min: 26558, max: 26558, variance: 0
        Probe samples:
          Packet Rx Timestamp       Measured Delay (nsec)
          Jan 20 2020 18:44:19.198        26558375
        Liveness Detection: Disabled
```

```
Router# show performance-measurement history probe sr-policy
Mon Jan 20 18:46:55.445 PST

------------------------------------------------------------------------------
0/0/CPU0
------------------------------------------------------------------------------

SR Policy name: srte_c_10_ep_192.168.0.4
  Color                       : 10
  Endpoint                    : 192.168.0.4

  Candidate-Path:
    Preference                : 100
    Protocol-origin           : Configured
    Discriminator             : 100
    Delay-Measurement history (uSec):
      Probe Start Timestamp     Pkt(TX/RX)   Average       Min       Max
      Jan 20 2020 18:46:34.174         9/9     26880     26684     27070
      Jan 20 2020 18:46:19.174         9/9     26899     26822     27004
      Jan 20 2020 18:46:04.173         9/9     26813     26571     27164
      Jan 20 2020 18:45:49.172         9/9     26985     26713     27293
      Jan 20 2020 18:45:34.172         9/9     26744     26557     27005
      Jan 20 2020 18:45:19.171         9/9     26740     26435     27093
     Jan 20 2020 18:45:04.171         9/9     27115     26938     27591
      Jan 20 2020 18:44:49.171         9/9     26878     26539     27143
      Jan 20 2020 18:44:34.171         9/9     26824     26562     27265
      Jan 20 2020 18:44:19.170         9/9     26944     26558     27422
      Jan 20 2020 18:44:06.543         9/9     45218     26512     82600

    Segment-List              : R4
      16004
      Delay-Measurement history (uSec):
        Probe Start Timestamp     Pkt(TX/RX)   Average       Min       Max
        Jan 20 2020 18:46:34.174         9/9     26880     26684     27070
        Jan 20 2020 18:46:19.174         9/9     26899     26822     27004
        Jan 20 2020 18:46:04.173         9/9     26813     26571     27164
        Jan 20 2020 18:45:49.172         9/9     26985     26713     27293
        Jan 20 2020 18:45:34.172         9/9     26744     26557     27005
        Jan 20 2020 18:45:19.171         9/9     26740     26435     27093
        Jan 20 2020 18:45:04.171         9/9     27115     26938     27591
        Jan 20 2020 18:44:49.171         9/9     26878     26539     27143
        Jan 20 2020 18:44:34.171         9/9     26824     26562     27265
        Jan 20 2020 18:44:19.170         9/9     26944     26558     27422
        Jan 20 2020 18:44:06.543         9/9     45218     26512     82600
```

```
  Atomic path:
    Hops                  : 127.0.0.0
    Delay-Measurement history (uSec):
      Probe Start Timestamp     Pkt(TX/RX)   Average       Min       Max
      Jan 20 2020 18:46:34.174         3/3     26927     26747     27070
      Jan 20 2020 18:46:19.174         3/3     26982     26970     27004
      Jan 20 2020 18:46:04.173         3/3     26895     26647     27164
      Jan 20 2020 18:45:49.172         3/3     27054     26764     27293
      Jan 20 2020 18:45:34.172         3/3     26801     26694     27005
      Jan 20 2020 18:45:19.171         3/3     26807     26524     27093
      Jan 20 2020 18:45:04.171         3/3     27226     26938     27591
      Jan 20 2020 18:44:49.171         3/3     26976     26644     27143
      Jan 20 2020 18:44:34.171         3/3     26880     26679     27265
      Jan 20 2020 18:44:19.170         3/3     26994     26630     27422
      Jan 20 2020 18:44:06.543         3/3     45407     26629     82600

  Atomic path:
    Hops                  : 127.0.0.1
    Delay-Measurement history (uSec):
      Probe Start Timestamp     Pkt(TX/RX)   Average       Min       Max
      Jan 20 2020 18:46:34.174         3/3     26865     26705     26988
      Jan 20 2020 18:46:19.174         3/3     26846     26822     26881
      Jan 20 2020 18:46:04.173         3/3     26787     26581     26939
      Jan 20 2020 18:45:49.172         3/3     26954     26728     27180
      Jan 20 2020 18:45:34.172         3/3     26724     26577     26957
      Jan 20 2020 18:45:19.171         3/3     26705     26452     27032
      Jan 20 2020 18:45:04.171         3/3     27043     26972     27124
      Jan 20 2020 18:44:49.171         3/3     26848     26550     27062
      Jan 20 2020 18:44:34.171         3/3     26800     26562     27204
      Jan 20 2020 18:44:19.170         3/3     26927     26576     27327
      Jan 20 2020 18:44:06.543         3/3     45128     26521     81961

  Atomic path:
    Hops                  : 192.168.0.4
    Delay-Measurement history (uSec):
      Probe Start Timestamp     Pkt(TX/RX)   Average       Min       Max
      Jan 20 2020 18:46:34.174         3/3     26848     26684     26967
      Jan 20 2020 18:46:19.174         3/3     26871     26833     26913
      Jan 20 2020 18:46:04.173         3/3     26759     26571     26876
      Jan 20 2020 18:45:49.172         3/3     26947     26713     27163
      Jan 20 2020 18:45:34.172         3/3     26708     26557     26939
      Jan 20 2020 18:45:19.171         3/3     26708     26435     27075
      Jan 20 2020 18:45:04.171         3/3     27078     27016     27138
      Jan 20 2020 18:44:49.171         3/3     26812     26539     27043
      Jan 20 2020 18:44:34.171         3/3     26793     26582     27181
      Jan 20 2020 18:44:19.170         3/3     26911     26558     27308
      Jan 20 2020 18:44:06.543         3/3     45119     26512     81956
```

```
Router# show performance-measurement counters sr-policy name srte_c_10_ep_192.168.0.4
Mon Jan 20 18:47:55.499 PST

------------------------------------------------------------------------------
0/0/CPU0
------------------------------------------------------------------------------
SR Policy name: srte_c_10_ep_192.168.0.4
  Candidate-Path:
    Instance                  : 2
    Preference                : 100
    Protocol-origin           : Configured
    Discriminator             : 100
    Packets:
      Total sent                              : 141
      Total received                          : 141
```

```
Errors:
  Total sent errors                        : 0
  Total received errors                    : 0
Probes:
  Total started                            : 16
  Total completed                          : 15
  Total incomplete                         : 0
  Total advertisements                     : 2
Segment-List              : R4
  16004
  Packets:
    Total sent                             : 141
    Total received                         : 141
  Errors:
    Total sent errors                      : 0
    Total received errors                  : 0
  Probes:
    Total started                          : 16
    Total completed                        : 15
    Total incomplete                       : 0
    Total advertisements                   : 2
```



CHAPTER 11

# Configure Topology-Independent Loop-Free Alternate (TI-LFA)

***Table 43: Feature History Table***

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.

- Classic Loop-Free Alternate (LFA) is topology dependent, and therefore cannot protect all destinations in all networks. A limitation of LFA is that, even if one or more LFAs exist, the optimal LFA may not always be provided.
- Remote LFA (RLFA) extends the coverage to 90-95% of the destinations, but it also does not always provide the most desired repair path. RLFA also adds more operational complexity by requiring a targeted LDP session to the RLFAs to protect LDP traffic.

TI-LFA provides a solution to these limitations while maintaining the simplicity of the IPFRR solution.

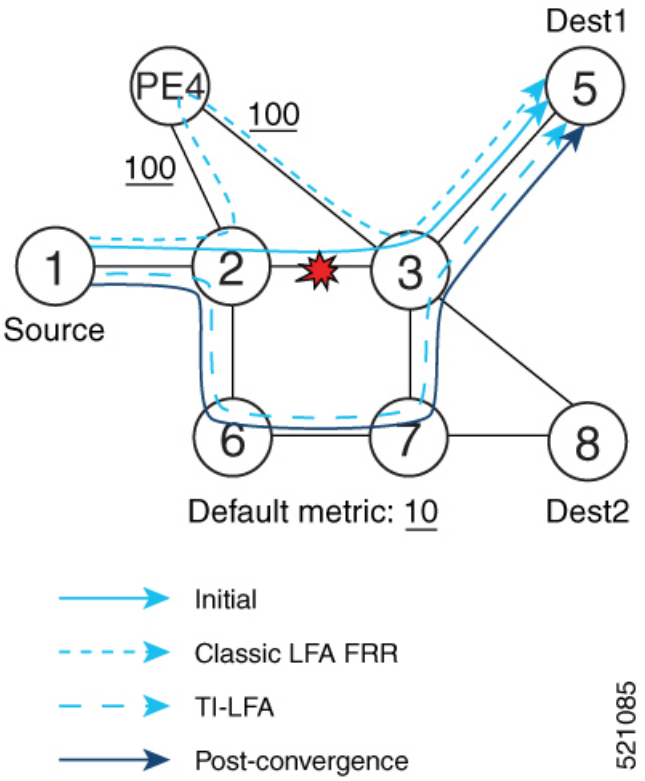
The goal of TI-LFA is to reduce the packet loss that results while routers converge after a topology change due to a link or node failure. Rapid failure repair (< 50 msec) is achieved through the use of pre-calculated backup paths that are loop-free and safe to use until the distributed network convergence process is completed.

The optimal repair path is the path that the traffic will eventually follow after the IGP has converged. This is called the post-convergence path. This path is preferred for the following reasons:

- Optimal for capacity planning — During the capacity-planning phase of the network, the capacity of a link is provisioned while taking into consideration that such link with be used when other links fail.
- Simple to operate — There is no need to perform a case-by-case adjustments to select the best LFA among multiple candidate LFAs.
- Fewer traffic transitions — Since the repair path is equal to the post-convergence path, the traffic switches paths only once.

The following topology illustrates the optimal and automatic selection of the TI-LFA repair path.

*Figure 30: TI-LFA Repair Path*




Node 2 protects traffic to destination Node 5.

With classic LFA, traffic would be steered to Node 4 after a failure of the protected link. This path is not optimal, since traffic is routed over edge node Node 4 that is connected to lower capacity links.

TI-LFA calculates a post-convergence path and derives the segment list required to steer packets along the post-convergence path without looping back.

In this example, if the protected link fails, the shortest path from Node2 to Node5 would be:

Node2 → Node6 → Node7 → Node3 → Node5

Node7 is the PQ-node for destination Node5. TI-LFA encodes a single segment (prefix SID of Node7) in the header of the packets on the repair path.

### TI-LFA Protection Types

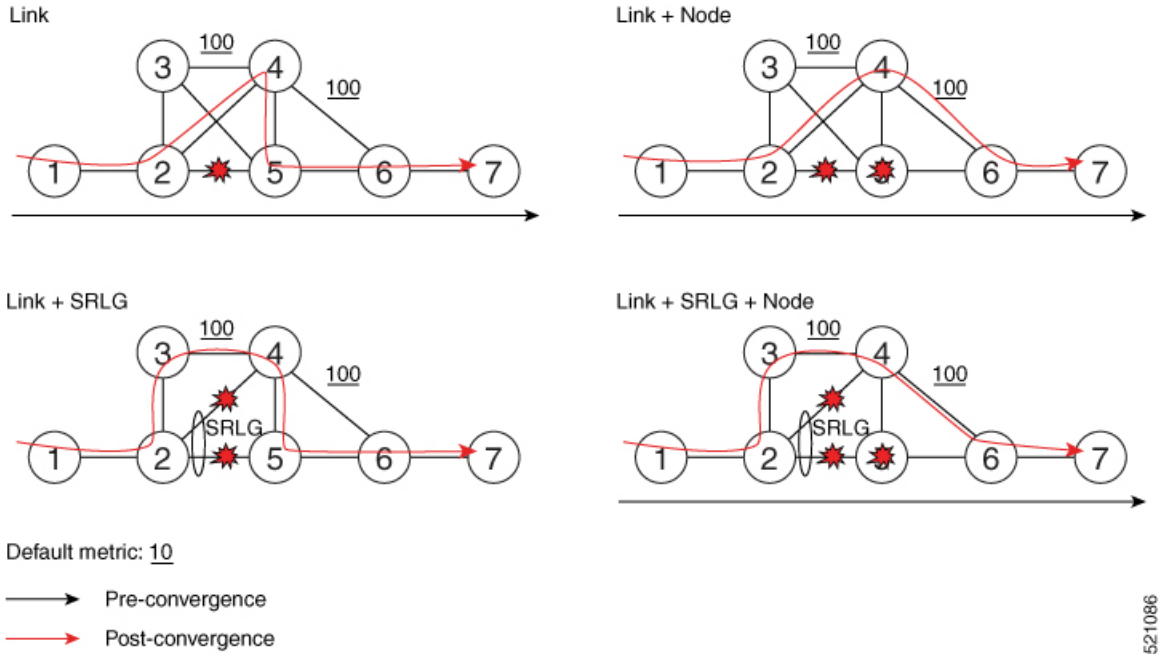
TI-LFA supports the following protection:

- Link protection — The link is excluded during the post-convergence backup path calculation.
- Node protection — The neighbor node is excluded during the post convergence backup path calculation.
- Shared Risk Link Groups (SRLG) protection — 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.

The following example illustrates the link, node, and SRLG protection types. In this topology, Node2 applies different protection models to protect traffic to Node7.

***Figure 31: TI-LFA Protection Types***




- Limitations, on page 373
- Usage guidelines and limitations for TI-LFA, on page 373
- Configuring TI-LFA for IS-IS, on page 375
- Configuring TI-LFA for OSPF, on page 376
- TI-LFA Node and SRLG Protection: Examples, on page 378
- Configuring Global Weighted SRLG Protection, on page 379
- SR-MPLS over GRE as TI-LFA Backup Path, on page 381

# Limitations

Only two backup labels are supported.

# Usage guidelines and limitations for TI-LFA

### Guidelines

- IGP directly programs a TI-LFA backup path requiring 3 or fewer labels, including the label of the protected destination prefix.

### Limitations

- The platform does not support programming of TI-LFA backup paths requiring more than 3 labels.

| TI-LFA Functionality | IS-IS[1] | OSPFv2 |
|---|---|---|
| ***Protected Traffic Types*** | | |
| Protection for SR labeled traffic | Supported | Supported |
| Protection of IPv4 unlabeled traffic | Supported (IS-ISv4) | Supported |
| Protection of IPv6 unlabeled traffic | Supported (IS-ISv6) | N/A |
| ***Protection Types*** | | |
| Link Protection | Supported | Supported |
| Node Protection | Supported | Supported |
| Local SRLG Protection | Supported | Supported |
| Weighted Remote SRLG Protection | Supported | Supported |
| Line Card Disjoint Protection | Supported | Unsupported |
| ***Interface Types*** | | |
| Ethernet Interfaces | Supported | Supported |
| TI-LFA with L3VPN | Supported | Supported |
| Ethernet Bundle Interfaces | Supported | Supported |
| TI-LFA over GRE Tunnel as Protecting Interface | Supported | Supported |
| ***Additional Functionality*** | | |
| Maximum number of labels that can be pushed on the backup path (including the label of the protected prefix) | 3 | 3 |
| BFD-triggered | Supported | Supported |
| BFDv6-triggered | Supported | N/A |
| Prefer backup path with lowest total metric | Supported | Supported |
| Prefer backup path from ECMP set | Supported | Supported |
| Prefer backup path from non-ECMP set | Supported | Supported |
| Load share prefixes across multiple backups paths | Supported | Supported |
| Limit backup computation up to the prefix priority | Supported | Supported |

[1] Unless specified, IS-IS support is IS-ISv4 and IS-ISv6

# 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:

- Routers are configured with IS-IS.
- Segment routing for IS-IS is configured. See Enabling Segment Routing for IS-IS Protocol, on page 105.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>**Note** You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command. |
| **Step 3** | **interface** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis)# interface GigabitEthernet0/0/0/1` | Enters interface configuration mode. |
| **Step 4** | **address-family ipv4** [**unicast**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast` | Specifies the IPv4 address family, and enters router address family configuration mode. |
| **Step 5** | **fast-reroute per-prefix**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if-af)#` | Enables per-prefix fast reroute. |

| | Command or Action | Purpose |
|---|---|---|
| | `fast-reroute per-prefix` | |
| **Step 6** | **fast-reroute per-prefix ti-lfa**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if-af)# 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*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tie-breaker 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.<br>**Note** The same attribute cannot be configured more than once on an interface.<br>**Note** For IS-IS, TI-LFA node protection and SRLG protection can be configured on the interface or the instance. |

TI-LFA has been successfully configured for segment routing.

# Configuring TI-LFA for OSPF

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

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

### Before you begin

Ensure that the following topology requirements are met:

- Routers are configured with OSPF.
- Segment routing for OSPF is configured. See Enabling Segment Routing for OSPF Protocol, on page 131.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router ospf** *process-name*<br>**Example:**<br>`RP/0/RP0/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*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# area 1` | Enters area configuration mode. |
| **Step 4** | **interface** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar)# interface GigabitEthernet0/0/0/1` | Enters interface configuration mode. |
| **Step 5** | **fast-reroute per-prefix**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix` | Enables per-prefix fast reroute. |
| **Step 6** | **fast-reroute per-prefix ti-lfa**<br>**Example:**<br>`RP/0/RP0/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*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix tie-breaker 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 higher the *priority* value, the higher the priority of the rule. Link protection always has a lower priority than node or SRLG protection.<br>**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 200
    fast-reroute per-prefix tiebreaker srlg-disjoint index 100
```

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

# Configuring Global Weighted SRLG Protection

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

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

### Configuration Examples: Global Weighted SRLG Protection

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

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

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

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
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)# 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)# 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)# 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 how to configure the global weighted SRLG protection feature with remote SRLG flooding.The configuration includes local and remote router configuration. On the local router, the global weighted SRLG protection is enabled by using the **fast-reroute per-prefix srlg-protection weighted-global** command. In the remote router configuration, you can control the SRLG value flooding by using the **advertise application lfa link-attributes srlg** command. You should also globally configure SRLG on the remote router.

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

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

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)# name group value 100
RP/0/RP0/CPU0:router(config-srlg)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-af)# advertise application lfa link-attributes srlg
```

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

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg)# name group value 100
RP/0/RP0/CPU0:router(config-srlg)# 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)# 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)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.1 next-hop ipv4
address 10.0.4.2
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.2 next-hop ipv4
address 10.0.4.1
```

# SR-MPLS over GRE as TI-LFA Backup Path

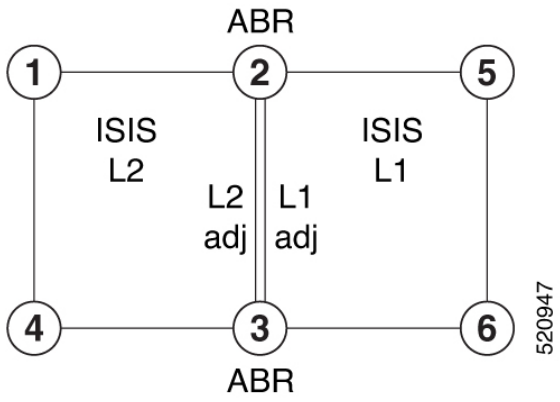
This feature allows the router (as ABR) to program a Generic Routing Encapsulation (GRE) tunnel as an outgoing interface for TI-LFA backup paths computed by the IGP in a Segment Routing network. Single-segment TI-LFA scenario is supported. In this scenario, the router pushes one extra label when programming the backup path.

**Note** GRE is a tunneling protocol that provides a simple generic approach to transport packets of one protocol over another protocol by means of encapsulation. See the *Configuring GRE Tunnels* chapter in the *Interface and Hardware Component Configuration Guide for Cisco NCS 540 Series Routers*.

### Multi-Level Network Topology

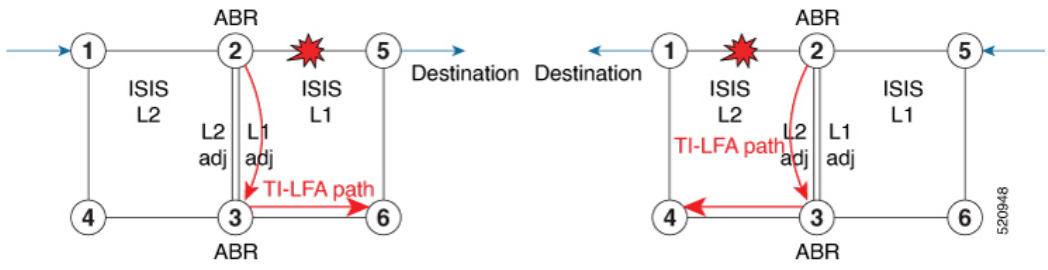
The following example shows a multi-level network topology with interconnecting links between ABRs.

**Note** This could also be a multi-instance network topology.




Two links between ABR 2 and ABR 3 are required, one in each IS-IS level. These links provide protection in each direction and ensure that there is always an alternate path inside the IGP domain.

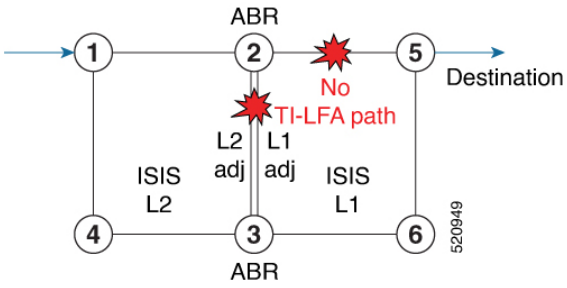
**Note** Alternatively, a single link with two logical sub-interfaces could be used between the ABRs.



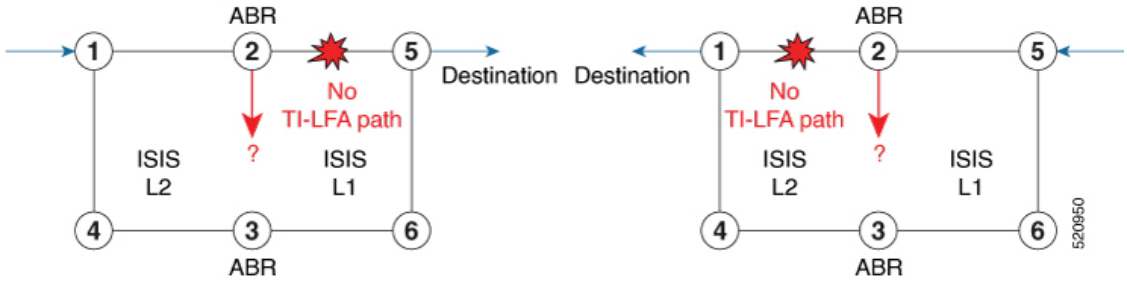
TI-LFA performs the backup path calculation inside the domain (process, level, or area) of the failed link.

For example, if the link between nodes 2 and 5 failed, the link between ABR 2 and 3 would create a TI-LFA path in L1 IS-IS level. If the link between nodes 1 and 2 failed, the link between ABR 2 and 3 would create a TI-LFA path in L2 IS-IS level.

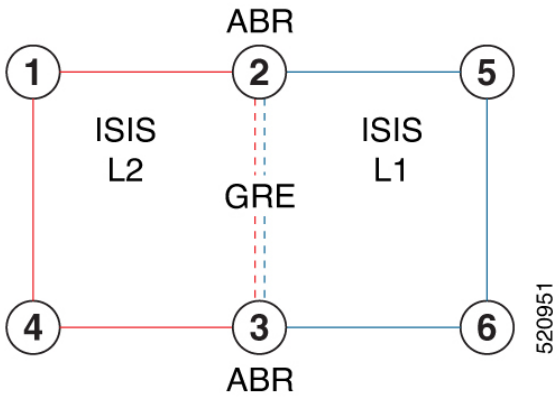
However, if the interconnecting link between ABRs are in the same Shared Risk Link Groups (SRLG) as other links inside the domain (for example, the link between Nodes 2 and 3 are in the same SRLG as link between Nodes 2 and 5), TI-LFA with local SRLG protection would not find an alternate path.



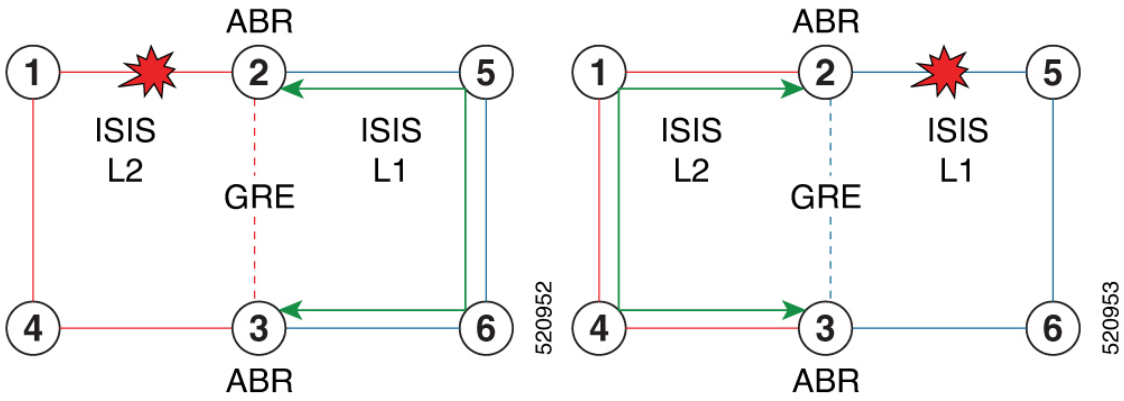

In cases where it is not feasible to provide interconnecting links between ABRs (for example, the ABR nodes might be in different locations with no connectivity options), TI-LFA will not be able to compute backup paths for all of the prefixes.




To address these issues, you can create a GRE tunnel in each domain, between the ABRs, which can be used as TI-LFA backup paths.

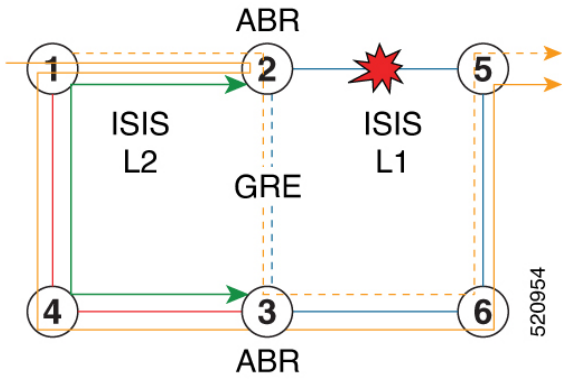



Now, if a link failure occurs in either IS-IS level (for example, between nodes 1 and 2 or between nodes 2 and 5), the path is protected by the GRE tunnel.

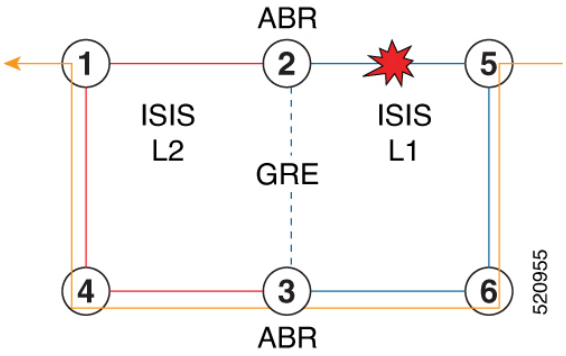



**Backup Path for Link Failure Between Nodes 2 and 5**

Traffic from node 1 is rerouted over the GRE tunnel TI-LFA backup path between ABR nodes 2 and 3.




Traffic flowing in the opposite direction, from node 5 to node 1, is simply routed over nodes 6-3-4 to node 1.




# Limitations

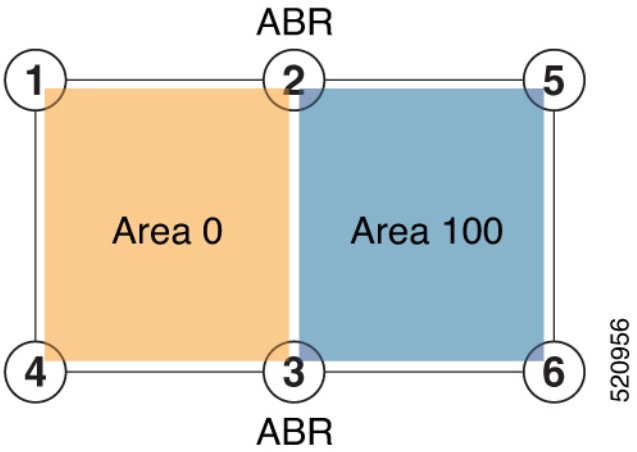
The following behaviors and limitations apply to the router when a GRE tunnel is programmed as backup interface for TI-LFA:

- The MPLS label of a protected prefix must be the same in the primary and backup paths (SWAP scenario)

- Single-segment TI-LFA is supported. In this scenario, the router pushes one extra label when programming the backup path. The total label stack is 2, including the primary label and backup label.
- Double-segment (or more) TI-LFA is not supported. In this scenario, the router pushes two or more extra labels when programming the backup path.
- GRE tunnel as a primary or backup path for an SR policy with TI-LFA protection is not supported.

# Example: SR-MPLS over GRE as TI-LFA Backup Path

The examples in this section use the following network topology:




### Configurations Without Interconnecting ABR Links

The following sample configurations show OSPF configurations for nodes 2, 3 and 5. Nodes 2 and 3 are ABRs between Area 0 and Area 100. There is no connection between the ABRs.

#### Configuration on ABR 2 for Area 0 and Area 100

```
router ospf 100
 router-id 2.2.2.2
 segment-routing mpls
 segment-routing forwarding mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 segment-routing sr-prefer
 area 0
  interface Loopback0
   prefix-sid index 2
  !
 !
  interface TenGigE0/0/1/10
   network point-to-point
  !
 !
 area 100
  interface TenGigE0/0/1/11
    network point-to-point
```

```
RP/0/RSP0/CPU0:ABR2# show ospf neighbor area-sorted
Fri Jul 19 09:43:59.328 UTC

Neighbors for OSPF 100
Area 0

Neighbor ID     Pri State    Dead Time Address         Up Time  Interface
10.1.1.1        1   FULL/  - 00:00:35  10.1.2.1        1d20h    Te0/0/1/10

Total neighbor count: 1

Area 100

Neighbor ID     Pri State    Dead Time Address         Up Time  Interface
5.5.5.5         1   FULL/  - 00:00:33  10.2.5.5        1d20h    Te0/0/1/11

Total neighbor count: 1
```

### Configuration on ABR 3 for Area 0 and Area 100

```
router ospf 100
 router-id 3.3.3.3
 segment-routing mpls
 segment-routing forwarding mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 segment-routing sr-prefer
 area 0
   interface Loopback0
   prefix-sid index 3
  !
  interface TenGigE0/0/0/9
  network point-to-point
  !
 !
 area 100
  interface TenGigE0/0/0/3
  network point-to-point
 !

RP/0/RSP0/CPU0:ABR3# show ospf neighbor area-sorted
Fri Jul 19 09:33:35.816 UTC

Neighbors for OSPF 100
Area 0

Neighbor ID     Pri State    Dead Time Address         Up Time  Interface
4.4.4.4         1   FULL/  - 00:00:36  10.3.4.4        2d17h    Te0/0/0/9

Total neighbor count: 1

Area 100

Neighbor ID     Pri State    Dead Time Address         Up Time  Interface
6.6.6.6         1   FULL/  - 00:00:36  10.3.6.6        2d19h    Te0/0/0/3

Total neighbor count: 1
```

### Configuration on Node 5

```
segment-routing mpls
 !
 set-attributes
  address-family ipv4
   sr-label-preferred
```

```
 !
 connected-prefix-sid-map
  address-family ipv4
   5.5.5.5/32 index 5 range 1
 !
interface TenGigabitEthernet0/0/26
 description ***Connected to ABR 2
 ip address 10.2.5.5 255.255.255.0
 ip ospf network point-to-point
 cdp enable
!
interface TenGigabitEthernet0/0/27
 description ***Connected to Node 6
 ip address 10.5.6.5 255.255.255.0
 ip ospf network point-to-point
 cdp enable

router ospf 100
 router-id 5.5.5.5
 segment-routing area 100 mpls
 segment-routing mpls
 fast-reroute per-prefix enable prefix-priority low
 fast-reroute per-prefix ti-lfa
 fast-reroute per-prefix ti-lfa area 100
 passive-interface default
 no passive-interface TenGigabitEthernet0/0/26
 no passive-interface TenGigabitEthernet0/0/27
 network 10.2.5.0 0.0.0.255 area 100
 network 10.5.5.0 0.0.0.255 area 100
 network 10.5.6.0 0.0.0.255 area 100
 network 5.5.5.5 0.0.0.0 area 100


RP/0/RSP0/CPU0:Node5# show ip ospf neighbor
Load for five secs: 4%/1%; one minute: 4%; five minutes: 4%
Time source is NTP, 09:50:51.417 UTC Fri Jul 19 2019

Neighbor ID   Pri   State      Dead Time   Address         Interface
6.6.6.6         0   FULL/  -   00:00:32    10.5.6.6        TenGigabitEthernet0/0/27
2.2.2.2         0   FULL/  -   00:00:36    10.5.2.5        TenGigabitEthernet0/0/26
```

### TI-LFA Fast Reroute Coverage on Node 5

The following output shows that this configuration provides only 52% TI-LFA fast reroute coverage on Node 5:

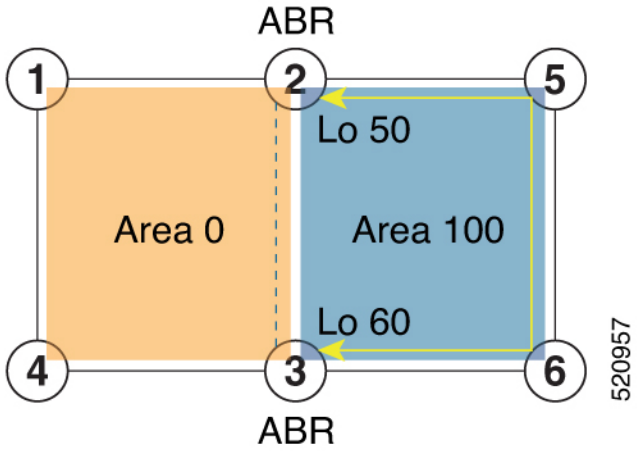
```
RP/0/RSP0/CPU0:Node5# show ip ospf fast-reroute prefix-summary
Load for five secs: 4%/1%; one minute: 4%; five minutes: 4%
Time source is NTP, 10:32:20.236 UTC Fri Jul 19 2019
            OSPF Router with ID (5.5.5.5) (Process ID 100)
                    Base Topology (MTID 0)
Area 100:
Interface        Protected    Primary paths    Protected paths Percent protected
                             All  High   Low   All  High   Low    All High  Low
Lo0                   Yes     0     0     0     0     0     0     0%   0%   0%
Te0/0/27              Yes     7     4     3     1     1     0    14%  25%   0%
Te0/0/26              Yes    10     5     5     8     4     4    80%  80%  80%

Area total:                  17     9     8     9     5     4    52%  55%  50%

Process total:               17     9     8     9     5     4    52%  55%  50%
```

### GRE Tunnel Configuration

The following examples show how to configure GRE tunnels between the ABRs in each area to provide TI-LFA backup paths for the Segment Routing network.

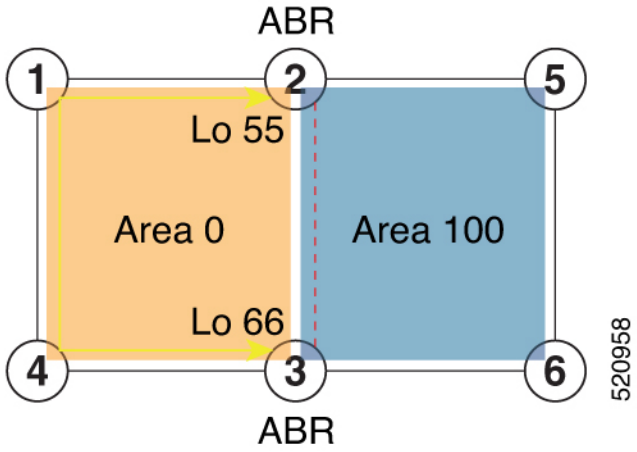
**GRE BLU** is configured in Area 0 using Loopback50 (on ABR2) and Loopback 60 (on ABR 3). These loopbacks are advertised in Area 100:

***Figure 32: GRE BLU***




**GRE RED** is configured in Area 100 using Loopback55 (on ABR2) and Loopback 66 (on ABR3). These loopbacks are advertised in Area 0:

***Figure 33: GRE RED***




### Configuration on ABR 2

```
interface Loopback0
 ipv4 address 2.2.2.2 255.255.255.255
!
interface Loopback50
 description Lo for GRE BLU
 ipv4 address 50.0.0.50 255.255.255.0
!
interface Loopback55
```

```
 description Lo for GRE RED
 ipv4 address 55.55.55.55 255.255.255.255
!
interface tunnel-ip5060
 description GRE virtual link for Area 0 BLU
 ipv4 address 66.3.2.2 255.255.255.0
 tunnel source Loopback50
 tunnel destination 60.0.0.60
!
interface tunnel-ip5566
 description GRE virtual link for Area 100 RED
 ipv4 address 100.3.2.2 255.255.255.0
 tunnel source Loopback55
 tunnel destination 66.66.66.66

router ospf 100
 router-id 2.2.2.2
 segment-routing mpls
 segment-routing forwarding mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 segment-routing sr-prefer
 area 0
  interface Loopback0
   prefix-sid index 2
  !
  interface Loopback55
   passive enable
  !
  interface tunnel-ip5060
   cost 1000
 !
  interface TenGigE0/0/1/10
   network point-to-point
  !
 !
 area 100
  interface Loopback50
   passive enable
  !
  interface tunnel-ip5566
   cost 1000
  !
  interface TenGigE0/0/1/11
    network point-to-point
```

**Note** In the above configuration, GRE tunnel-ip5060 belongs to area 0, but its source and destination addresses are advertised in area 100. This ensures disjointness between the GRE tunnel and the links in area 0 that it protects. The same applies to GRE tunnel-ip5566 which belongs to area 100 and its source and destination addresses are advertised in area 0.

A high cost is applied to the GRE tunnel interfaces so that they are used only as a backup path.

### Configuration on ABR 3

```
interface Loopback0
 ipv4 address 3.3.3.3 255.255.255.255
!
interface Loopback60
 description Lo for GRE BLU
```

```
 ipv4 address 60.0.0.60 255.255.255.0
!
interface Loopback66
 description Lo for GRE RED
 ipv4 address 66.66.66.66 255.255.255.255
!
interface tunnel-ip5060
 description GRE virtual link for Area 0 BLU
 ipv4 address 66.3.2.3 255.255.255.0
 tunnel source Loopback60
 tunnel destination 50.0.0.50
!
interface tunnel-ip5566
 description GRE virtual link for Area 100 RED
 ipv4 address 100.3.2.3 255.255.255.0
 tunnel source Loopback66
 tunnel destination 55.55.55.55

router ospf 100
 router-id 3.3.3.3
 segment-routing mpls
 segment-routing forwarding mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 segment-routing sr-prefer
 area 0
   interface Loopback0
   prefix-sid index 3
  !
  interface TenGigE0/0/0/9
  network point-to-point
  !
  interface Loopback66
   passive enable
  !
  interface tunnel-ip5060
   cost 1000
 !
 area 100
  interface TenGigE0/0/0/3
  network point-to-point
 !
  interface Loopback60
   passive enable
  !
  interface tunnel-ip5566
   cost 1000
```

**Note** In the above configuration, GRE tunnel-ip5060 belongs to area 0, but its source and destination addresses are advertised in area 100. This ensures disjointness between the GRE tunnel and the links in area 0 that it protects. The same applies to GRE tunnel-ip5566 which belongs to area 100 and its source and destination addresses are advertised in area 0.

A high cost is applied to the GRE tunnel interfaces so that they are used only as a backup path.

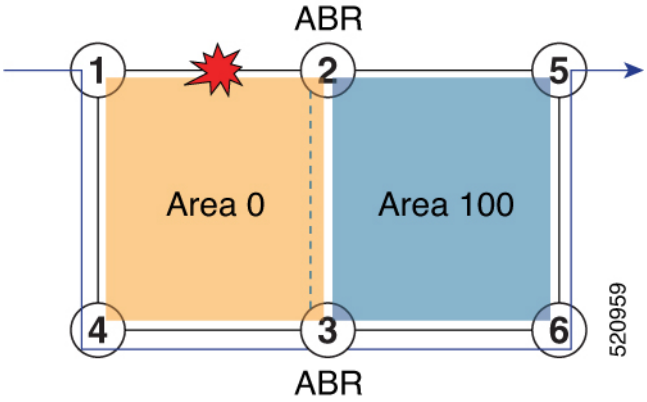
### TI-LFA Fast Reroute Coverage on Node 5 After GRE Tunnel Configuration

The following output shows that this configuration provides 100% TI-LFA fast reroute coverage on Node 5:

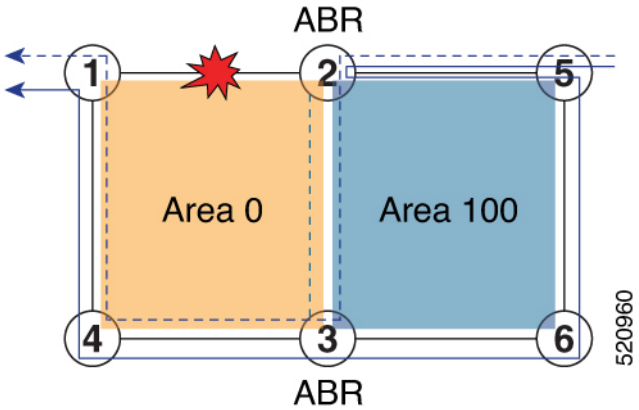
```
RP/0/RSP0/CPU0:Node5# show ip ospf fast-reroute prefix-summary
Load for five secs: 5%/1%; one minute: 4%; five minutes: 4%
Time source is NTP, 11:20:31.743 UTC Fri Jul 19 2019
            OSPF Router with ID (5.5.5.5) (Process ID 100)
                    Base Topology (MTID 0)
Area 100:
Interface        Protected    Primary paths     Protected paths Percent protected
                            All  High   Low    All  High   Low    All High  Low
Lo0                   Yes     0     0     0     0     0     0     0%   0%   0%
Te0/0/27              Yes     9     6     3     9     6     3   100% 100% 100%
Te0/0/26              Yes    11     6     5    11     6     5   100% 100% 100%

Area total:                  20    12     8    20    12     8   100% 100% 100%

Process total:               20    12     8    20    12     8   100% 100% 100%
```

### Traffic Flow with GRE Tunnel as TI-LFA Backup

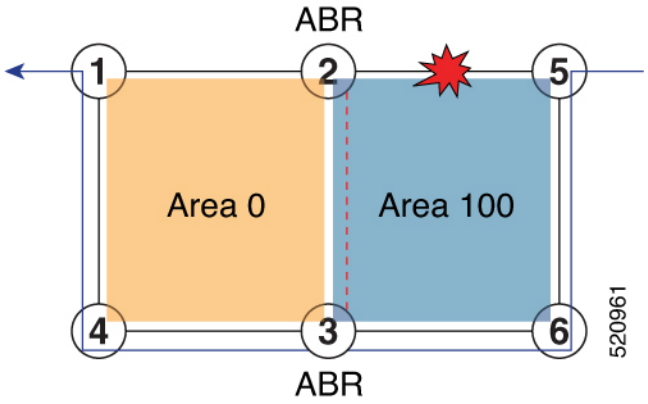
With a link failure between Node 1 and ABR 2, traffic flowing from Node 1 to Node 5 is simply routed through Nodes 4-3-6 to Node 5.



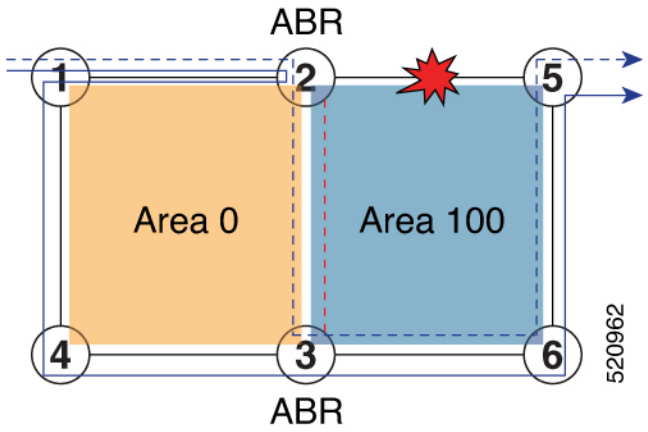

With GRE tunnel as TI-LFA backup, traffic flowing from Node 5 to Node 1 will be encapsulated at ABR2 and routing over the GRE tunnel.

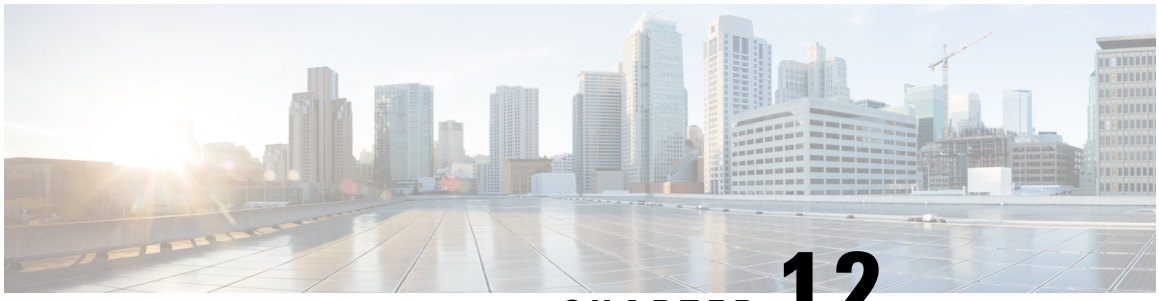



With a link failure between Node 5 and ABR 2, traffic flowing from Node 5 to Node 1 is simply routed through Nodes 6-3-4 to Node 1.




With GRE tunnel as TI-LFA backup, traffic flowing from Node 1 to Node 5 will be encapsulated at ABR2 and routing over the GRE tunnel.





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

IP hop-by-hop routing may induce microloops (uLoops) at any topology transition. Microloops are a day-one IP challenge. Microloops are brief packet loops that occur in the network following a topology change:

- Link down or up (remote or local)
- Metric increase or decrease (remote or local)

Microloops are caused by the non-simultaneous convergence of different nodes in the network. If a node converges and sends 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.

Segment Routing can be used to resolve the microloop problem. A router with the Segment Routing Microloop Avoidance feature detects if microloops are possible for a destination on the post-convergence path following a topology change associated with a remote link event.

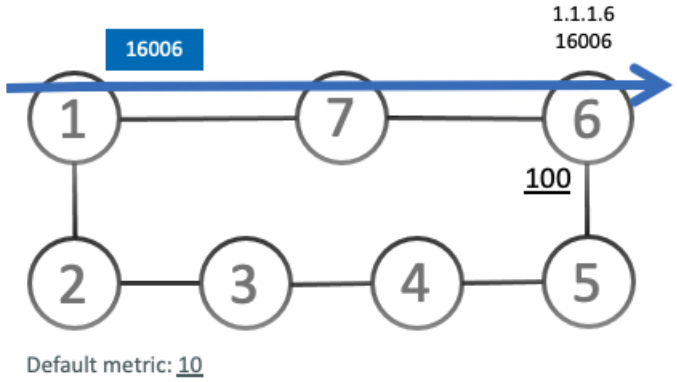
If a node determines that a microloop could occur on the new topology, the IGP computes a microloop-avoidant path by updating the forwarding table and temporarily (based on a RIB update delay timer) installing the SID-list imposition entries associated with the microloop-avoidant path for the destination. Traffic is steered to that destination loop-free.

After the RIB update delay timer expires, IGP updates the forwarding table and removes the microloop-avoidant SID list. Traffic now natively follows the post-convergence path.

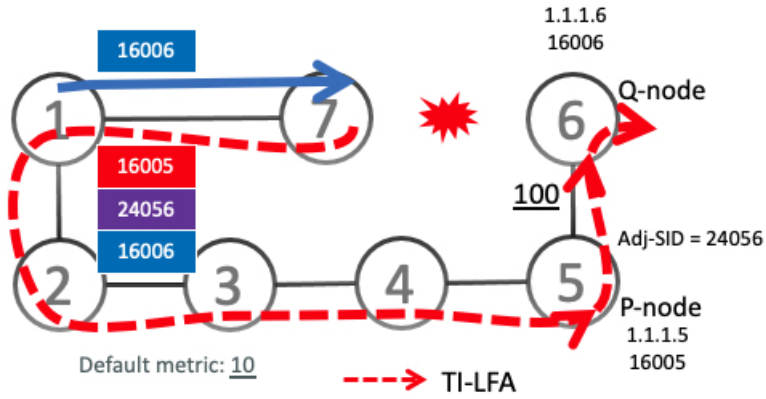
SR microloop avoidance is a local behavior and therefore not all nodes need to implement it to get the benefits.

In the topology below, microloops can occur after the failure of the link between Node6 and Node7.

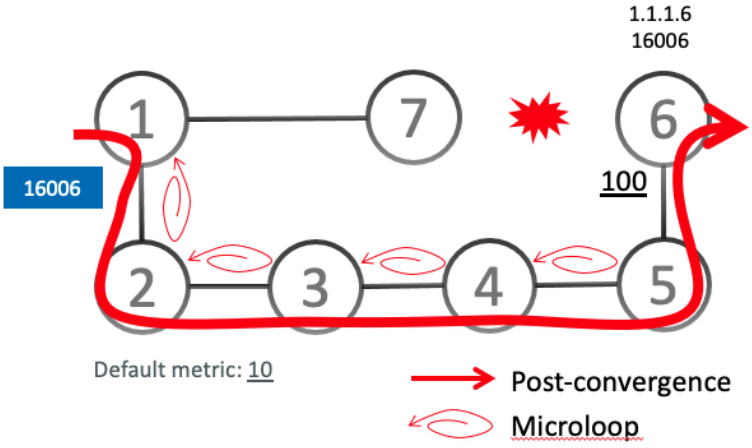
At steady state, Node1 sends traffic to node 6 (16006) via Node7. Node 7 is configured with TI-LFA to protect traffic to Node6.




TI-LFA on Node7 pre-computes a backup path for traffic to Node6 (prefix SID 16006) that will be activated if the link between Node7 and Node6 goes down. In this network, the backup path would steer traffic toward Node5 (prefix SID 16005) and then via link between Node5 and Node6 (adj-SID 24056). All nodes are notified of the topology change due to the link failure.



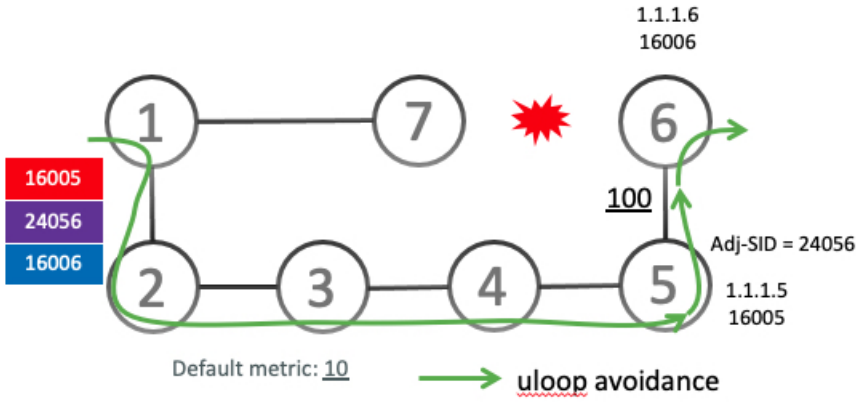

However, if nodes along the path do not converge at the same time, microloops can be introduced. For example, if Node2 converged before Node3, Node3 would send traffic back to Node2 as the shortest IGP path to Node6. The traffic between Node2 and Node3 creates a microloop.



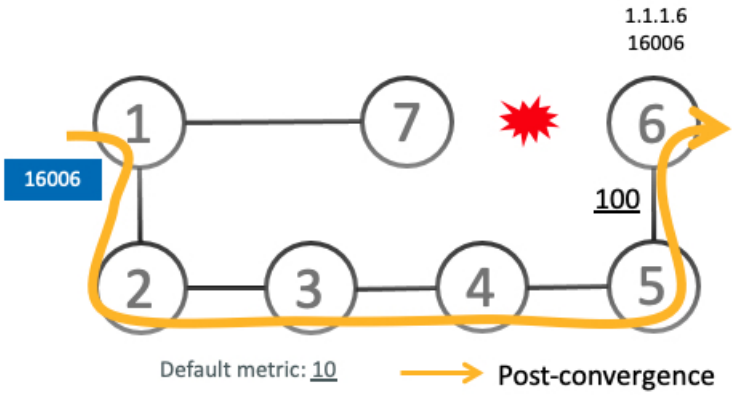
With microloop avoidance configured on Node1, a post-convergence path is computed and possible microloops on the post-convergence path for any destination are detected.

If microloops are possible on the post-convergence path to Node6, a microloop-avoidant path is constructed to steer the traffic to Node6 loop-free over the microloop-avoidant path {16005, 24056, 16006}.

Node1 updates the forwarding table and installs the SID-list imposition entries for those destinations with possible microloops, such as Node6. All nodes converge and update their forwarding tables, using SID lists where needed.




After the RIB update delay timer expires, the microloop-avoidant path is replaced with regular forwarding paths; traffic now natively follows the post-convergence path.




# Usage Guidelines and Limitations

# 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:

- Routers are configured with IS-IS.
- Segment routing for IS-IS is configured. See Enabling Segment Routing for IS-IS Protocol, on page 105.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>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** [ **unicast** ]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast` | Specifies the IPv4 address family and enters router address family configuration mode. |
| **Step 4** | **microloop avoidance segment-routing**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-af)# microloop avoidance segment-routing` | Enables Segment Routing Microloop Avoidance. |
| **Step 5** | **microloop avoidance rib-update-delay** *delay-time*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-af)# microloop avoidance rib-update-delay 3000` | Specifies the amount of time the node uses the microloop avoidance policy before updating its forwarding table. The *delay-time* is in milliseconds. The range is from 1-60000. The default value is 5000. |

# 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.

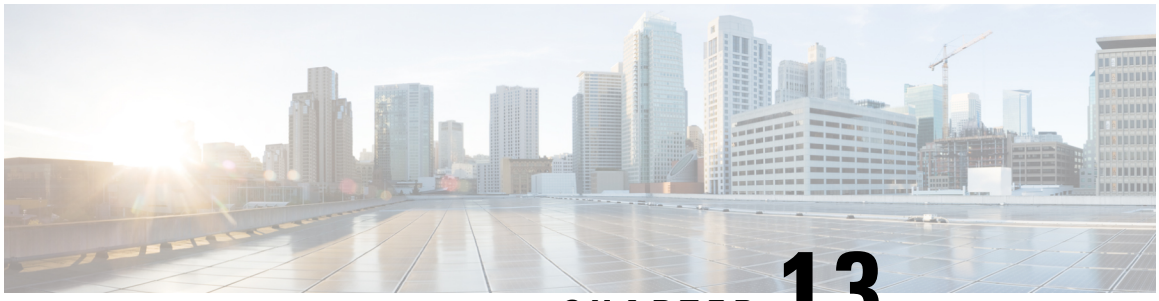
### Before you begin

Ensure that the following topology requirements are met:

- Routers are configured with OSPF.
- Segment routing for OSPF is configured. See Enabling Segment Routing for OSPF Protocol, on page 131.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **router ospf** *process-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router ospf 1` | Enables OSPF routing for the specified routing process, and places the router in router configuration mode. |
| **Step 3** | **microloop avoidance segment-routing**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# microloop avoidance segment-routing` | Enables Segment Routing Microloop Avoidance. |
| **Step 4** | **microloop avoidance rib-update-delay** *delay-time*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# microloop avoidance rib-update-delay 3000` | Specifies the amount of time the node uses the microloop avoidance path before updating its forwarding table. The *delay-time* is in milliseconds. The range is from 1-60000. The default value is 5000. |



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

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.

# Usage Guidelines and Restrictions

*Table 44: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Advertisement of SID-Mapping Entries Between IS-IS Levels | Release 7.3.1 | The Segment Routing Mapping Server (SRMS) is a key component of the interworking between LDP and segment routing, enabling SR-capable nodes to interwork with LDP nodes.<br><br>This release introduces support for SRMS SID-mapping entries to be advertised between IS-IS levels (for example, from Level 1 to Level 2-only and from Level 2 to Level 1), where previously, the mappings were advertised only within the same IS-IS level, but not between IS-IS levels. This feature simplifies and centralizes the deployment of SRMS by removing the requirement of having a mapping server for each IS-IS area. |

- 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 supports the advertisement of SID-mapping entries between IS-IS levels (for example, from L1 to L2-only and from L2 to L1). A mapping server is not required for each IS-IS area.

  For example, mapping entries learned from IS-IS Type Level-1 (intra-area) routers can be used to calculate prefix-SIDs for prefixes learned or advertised by IS-IS Type Level-2-only (backbone) routers.

  Use the **domain-wide** option to advertise the prefix-SID mappings between Level 1 and Level 2 IS-IS routers.
- 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 instance.
- Segment Routing Mapping Server does not support Virtual Routing and Forwarding (VRF) currently.

# Segment Routing and LDP Interoperability

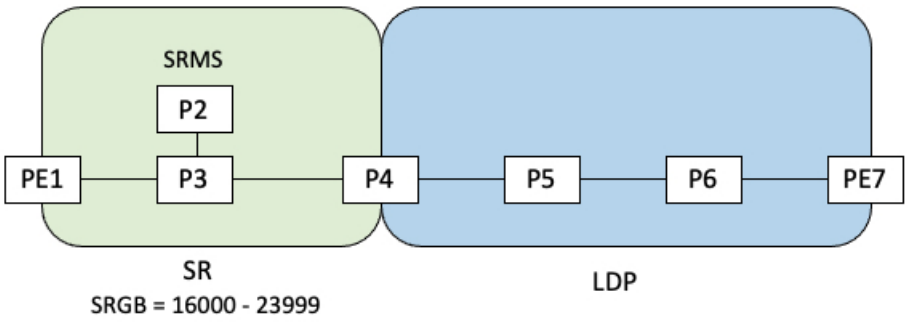
IGP provides mechanisms through which segment routing (SR) interoperate with label distribution protocol (LDP). The control plane of segment routing co-exists with LDP.

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 figure. This network is a mix of both LDP and SR-capable nodes.

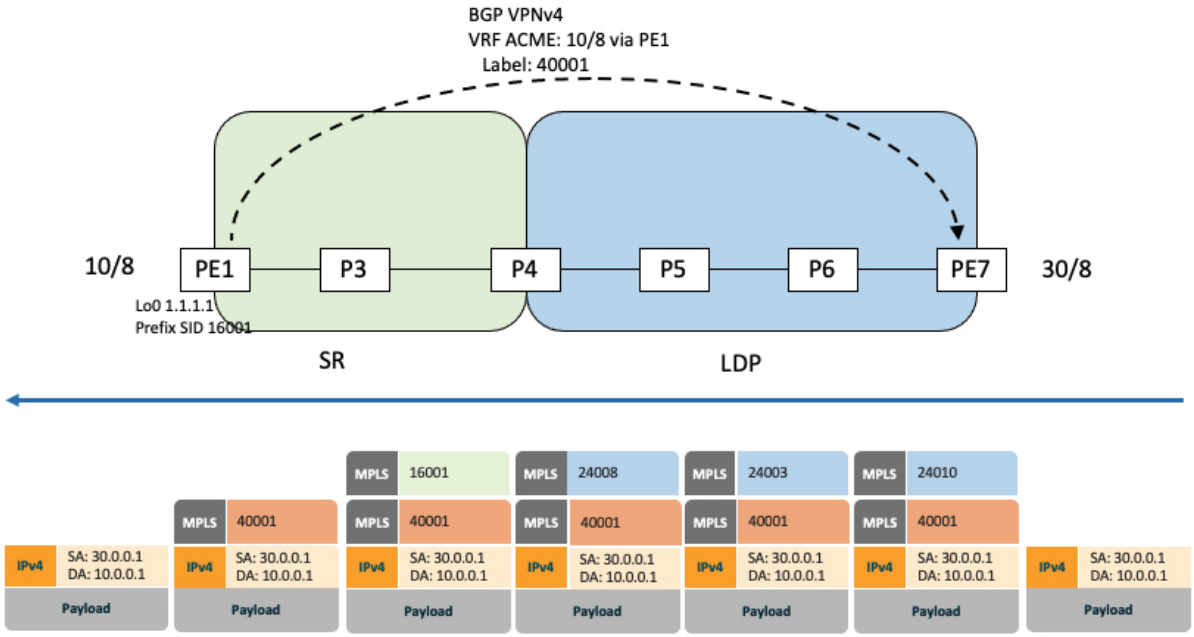



In this mixed network:

- Nodes PE1, P2, P3, and P4 are SR-capable
- Nodes P4, P5, P6, and PE7 are LDP-capable
- Nodes PE1, P2, P3, and P4 are configured with segment routing global block (SRGB) range of 16000 to 23999
- Nodes PE1, P2, P3, and P4 are configured with node segments of 16001, 16002, 16003, and 16004 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 Traffic Direction

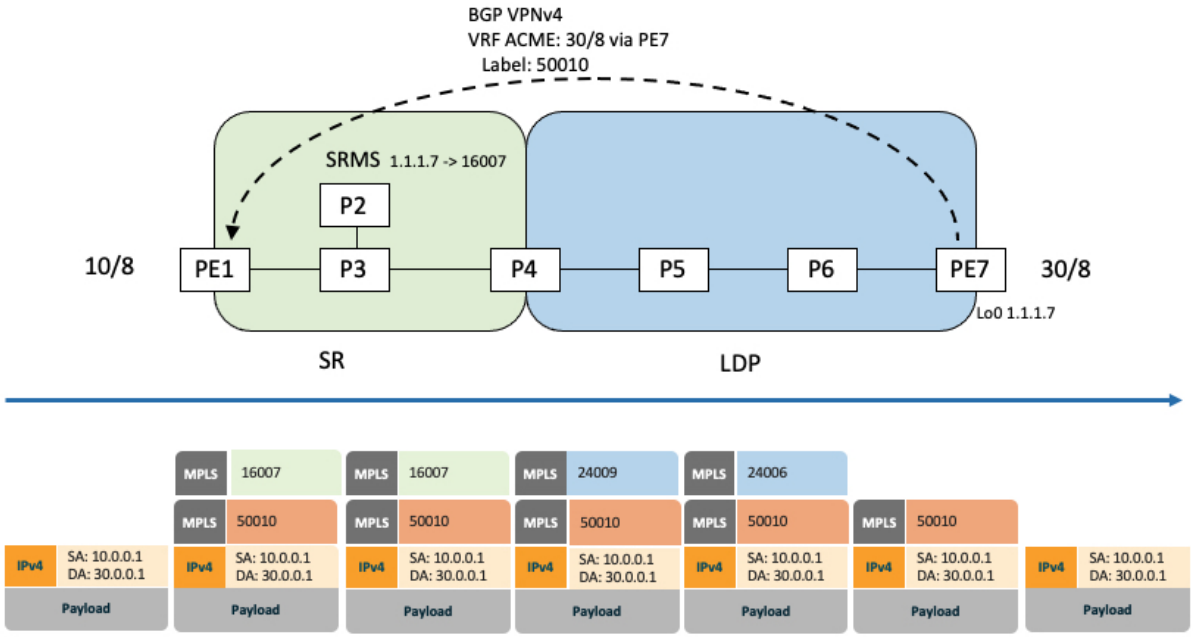



The traffic flow in the LDP-to-SR direction involves the following:

1. PE7 learns a service route with service label 40001 and BGP nhop PE1.
2. PE7 has an LDP label binding (24010) from the nhop P6 for the FEC PE1. PE7 forwards the packet to P6.
3. P6 has an LDP label binding (24003) from its nhop P5 for the FEC PE1. P6 forwards the packet to P5.
4. P5 has an LDP label binding (24008) from its nhop P4 for the FEC PE1. P5 forwards the packet to P4.
5. P4 does not have an LDP binding from its nhop P3 for the FEC PE1. But P4 has an SR node segment to the IGP route PE1. P4 forwards the packet to P3 and swaps its local LDP label (24008) for FEC PE1 by the equivalent node segment 16001. This process is called label merging.
6. P3 pops 16001, assuming PE1 has advertised its node segment 16001 with the penultimate-pop flag set and forwards to PE1.
7. PE1 receives the packet and processes the service label.

The end-to-end MPLS LSP is established from an LDP LSP from PE7 to P4 and the related node segment from P4 to PE1.

### SR-to-LDP Traffic Direction




Suppose that the operator configures P2 as a Segment Routing Mapping Server (SRMS) and advertises the mappings (1.1.1.7, 16007 for PE7). Because PE7 is non-SR capable, the operator configures that mapping 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 as if node segments were advertised by the nodes themselves.

The traffic flow in the SR to LDP direction involves the following:

1. PE1 learns a service route with service label 50010 and BGP nhop PE7.
2. PE1 has an SR label binding (16007) learned from the SRMS (P2) for PE7.
3. PE1 installs the node segment 16007 following the IGP shortest-path with nhop P3.
4. P3 swaps 16007 for 16007 and forwards to P4.
5. The nhop for P4 for the IGP route PE7 is non-SR capable, since P5 does not advertise the SR capability. However, P4 has an LDP label binding from that nhop for the same FEC (for example, LDP label 24009). P4 would then swap 16007 for 24009 and forward to P5. We refer to this process as label merging.
6. P5 swaps this label with the LDP label received from P6 (for example, LDP label 24006) and forwards to P6.
7. P6 pops the LDP label and forwards to PE7.
8. PE7 receives the packet and processes the service label.

The end-to-end MPLS LSP is established from an SR node segment from PE1 to P4 and an LDP LSP from P4 to PE7.

Observe that the capabilities provided by the SRMS are only required in the SR-to-LDP direction.

# Configuring Mapping Server

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **segment-routing**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# segment-routing` | Enables segment routing. |
| **Step 3** | **mapping-server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sr)# mapping-server` | Enables mapping server configuration mode. |
| **Step 4** | **prefix-sid-map**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sr-ms)# prefix-sid-map` | Enables prefix-SID mapping configuration mode.<br>**Note** Two-way prefix SID can be enabled directly under IS-IS or through a mapping server. |
| **Step 5** | **address-family ipv4** \| **ipv6**<br>**Example:**<br>This example shows the address-family for ipv4:<br>`RP/0/RP0/CPU0:router(config-sr-ms-map)# address-family ipv4`<br>This example shows the address-family for ipv6:<br>`RP/0/RP0/CPU0:router(config-sr-ms-map)# address-family ipv6` | Configures address-family for IS-IS. |
| **Step 6** | *ip-address*/*prefix-length* *first-SID-value* **range** *range* | Adds SID-mapping entries in the active local mapping policy. In the configured example: |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br><br>`RP/0/RP0/CPU0:router(config-sr-ms-map-af)# 10.1.1.1/32 10 range 200`<br>`RP/0/RP0/CPU0:router(config-sr-ms-map-af)# 20.1.0.0/16 400 range 300` | • 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<br>• 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. |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Verify information about the locally configured prefix-to-SID mappings.

**Note** Specify the address family for IS-IS.

```
RP/0/RP0/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/RP0/CPU0:router# show segment-routing mapping-server prefix-sid-map ipv4 detail
Prefix
20.1.1.0/24
    SID Index:      400
    Range:          300
    Last Prefix:    20.2.44.0/24
    Last SID Index: 699
    Flags:
10.1.1.1/32
    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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **router isis** *instance-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router isis 1` | Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.<br>• You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command. |
| **Step 2** | **address-family** { **ipv4** \| **ipv6** } [ **unicast** ]<br>**Example:**<br>The following is an example for ipv4 address family:<br>`RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast` | Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode. |
| **Step 3** | **segment-routing prefix-sid-map advertise-local** [**domain-wide**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-af)# segment-routing prefix-sid-map advertise-local`<br>`RP/0/RP0/CPU0:router(config-isis-af)# segment-routing prefix-sid-map advertise-local domain-wide` | Configures IS-IS to advertise locally configured prefix-SID mappings. Use the **domain-wide** option to advertise the prefix-SID mappings between IS-IS Level 1 and Level 2 routers. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Verify IS-IS prefix-SID mapping advertisement and TLV.

```
RP/0/RP0/CPU0:router# show isis database verbose

<...removed...>

 SID Binding:  10.1.1.1/32 F:0 M:0 S:0 D:0 A:0 Weight:0 Range:200
    SID: Start:10, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0
 SID Binding:  20.1.1.0/24 F:0 M:0 S:0 D:0 A:0 Weight:0 Range:300
    SID: Start:400, Algorithm:0, R:0 N:0 P:0 E:0 V:0 L:0
```

# Configure Mapping Advertisement for OSPF

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **router ospf** *process-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router ospf 1` | Enables OSPF routing for the specified routing instance, and places the router in router configuration mode. |
| **Step 2** | **segment-routing prefix-sid-map advertise-local**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospf)# segment-routing prefix-sid-map advertise-local` | Configures OSPF to advertise locally configured prefix-SID mappings. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| | | **end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

Verify OSP prefix-SID mapping advertisement and TLV.

```
RP/0/RP0/CPU0:router# show ospf database opaque-area

<...removed...>

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

      SID sub-TLV: Length: 8
        Flags     : 0x60
        MTID      : 0
        Algo      : 0
        SID Index : 10
```

# 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.

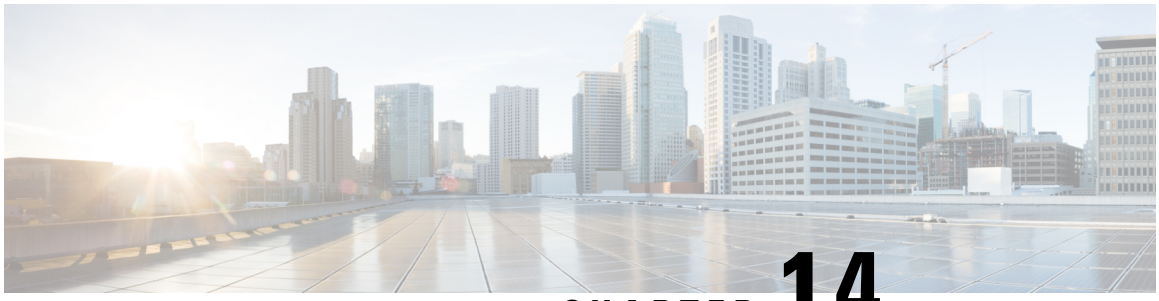
You can re-enable the mapping client functionality by using the **segment-routing prefix-sid-map receive** command.

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

```
RP/0/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-af)# segment-routing prefix-sid-map receive
```

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

```
RP/0/RP0/CPU0:router(config)# router ospf 1
RP/0/RP0/CPU0:router(config-ospf)# segment-routing prefix-sid-map receive disable
RP/0/RP0/CPU0:router(config-ospf)# commit
```



CHAPTER 14

# Enabling Segment Routing Flexible Algorithm

Segment Routing Flexible Algorithm allows operators to customize IGP shortest path computation according to their own needs. An operator can assign custom SR prefix-SIDs to realize forwarding beyond link-cost-based SPF. As a result, Flexible Algorithm provides a traffic engineered path automatically computed by the IGP to any destination reachable by the IGP.

The SR architecture associates prefix-SIDs to an algorithm which defines how the path is computed. Flexible Algorithm allows for user-defined algorithms where the IGP computes paths based on a user-defined combination of metric type and constraint.

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



# 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 and OSPF.

## Flexible Algorithm Definition

Many possible constraints may be used to compute a path over a network. Some networks are deployed with multiple planes. A simple form of constraint may be to use a particular plane. A more sophisticated form of constraint 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 a Flexible Algorithm.

# Flexible Algorithm Membership

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
- Exclude SRLG constraint

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 Link Attribute Advertisement

Various link attributes may be used during the Flexible Algorithm path calculation. For example, include or exclude rules based on link affinities can be part of the Flexible Algorithm definition, as defined in RFC9350 (IGP Flexible Algorithm).

Link attribute advertisements used during Flexible Algorithm calculation must use the Application-Specific Link Attribute (ASLA) advertisements, as defined in RFC8919 (IS-IS) and RFC8920 (OSPF). In the case of IS-IS, if the L-Flag is set in the ASLA advertisement, then legacy advertisements (IS-IS Extended Reachability TLV) are used instead.

# 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 a 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

*Table 45: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| OSPF: Microloop Avoidance for Flexible Algorithm | Release 7.4.1 | This feature extends the current OSPF Flexible Algorithm functionality to support Microloop Avoidance. |

*Table 46: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| OSPF: Microloop Avoidance for Flexible Algorithm | Release 7.3.2 | This feature extends the current OSPF Flexible Algorithm functionality to support Microloop Avoidance. |
| OSPF: TI-LFA for Flexible Algorithm | Release 7.3.1 | This feature extends the current OSPF Flexible Algorithm functionality to support TI-LFA. |

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 Flexible Algorithm is used.

The router uses the following rules to prune links from the topology during the Flexible Algorithm computation:

- All nodes that don't advertise support for Flexible Algorithm are pruned from the topology.
- Affinities:
  - Check if any exclude affinity rule is part of the Flexible Algorithm Definition. If such exclude rule exists, check if any color that is part of the exclude rule is also set on the link. If such a color is set, the link must be pruned from the computation.
  - Check if any include-any affinity rule is part of the Flexible Algorithm Definition. If such include-any rule exists, check if any color that is part of the include-any rule is also set on the link. If no such color is set, the link must be pruned from the computation.
  - Check if any include-all affinity rule is part of the Flexible Algorithm Definition. If such include-all rule exists, check if all colors that are part of the include-all rule are also set on the link. If all such colors are not set on the link, the link must be pruned from the computation

**Note** See Flexible Algorithm Affinity Constraint.

- If the Flexible Algorithm definition includes an "exclude SRLG" rule, then all links that are part of such SRLG are pruned from the topology.

**Note** See Flexible Algorithm with Exclude SRLG Constraint, on page 422.

- Router uses the metric that is part of the Flexible Algorithm definition. If the metric isn't advertised for the particular link, the link is pruned from the topology.

Loop Free Alternate (LFA) paths, TI-LFA backup paths, and Microloop Avoidance paths for particular Flexible Algorithm are computed using the same constraints as the calculation of the primary paths for such Flexible Algorithm. These paths use Prefix-SIDs advertised specifically for such Flexible Algorithm in order to enforce a backup or microloop avoidance path.

### Configuring Microloop Avoidance for Flexible Algorithm

By default, Microloop Avoidance per Flexible Algorithm instance follows Microloop Avoidance configuration for algo-0. For information about configuring Microloop Avoidance, see Configure Segment Routing Microloop Avoidance, on page 393.

You can disable Microloop Avoidance for Flexible Algorithm using the following commands:

```
router isis instance flex-algo algo microloop avoidance disable

router ospf process flex-algo algo microloop avoidance disable
```

### Configuring LFA / TI-LFA for Flexible Algorithm

By default, LFA/TI-LFA per Flexible Algorithm instance follows LFA/TI-LFA configuration for algo-0. For information about configuring TI-LFA, see Configure Topology-Independent Loop-Free Alternate (TI-LFA), on page 371.

You can disable TI-LFA for Flexible Algorithm using the following commands:

```
router isis instance flex-algo algo fast-reroute disable

router ospf process flex-algo algo fast-reroute disable
```

# 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.

# Flexible Algorithm Prefix-SID Redistribution

*Table 47: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Flexible Algorithm Prefix-SID Redistribution for External Route Propagation | Release 7.5.2 | You can now propagate flexible algorithm prefix-SIDs and their algorithm-specific metric between different IGP domains, such as OSPF to IS-IS RIP to OSPF. With this functionality enabling interdomain traffic engineering, you can export flexible algorithm labels from the OSPF domain to other domains and import the labels from other domains into OSPF.<br>The **show ospf route flex-algo** command has been modified to include additional attributes to indicate the external routes. |

Prefix redistribution from IS-IS to another IS-IS instance or protocol was limited to SR algorithm 0 (regular SPF) prefix SIDs; SR algorithm 1 (Strict SPF) and SR algorithms 128-255 (Flexible Algorithm) prefix SIDs were not redistributed along with the prefix. The Segment Routing IS-IS Flexible Algorithm Prefix SID Redistribution feature allows redistribution of strict and flexible algorithms prefix SIDs from IS-IS to another IS-IS instance or protocols. This feature is enabled automatically when you configure redistribution of IS-IS Routes with strict or flexible algorithm SIDs.

### Configuration Example

The following example shows how to configure redistribute and flexible algorithm to enable external routes.

```
RP/0/RP0/CPU0:ios(config)#router ospf 1
RP/0/RP0/CPU0:ios(config-ospf)#segment-routing mpls
RP/0/RP0/CPU0:ios(config-ospf)#segment-routing forwarding mpls
RP/0/RP0/CPU0:ios(config-ospf)#redistribute isis 2 route-policy loopback-type
RP/0/RP0/CPU0:ios(config-ospf)#flex-algo 240
RP/0/RP0/CPU0:ios(config-ospf-flex-algo)#metric-type delay
RP/0/RP0/CPU0:ios(config-ospf-flex-algo)#prefix-metric
RP/0/RP0/CPU0:ios(config-ospf-flex-algo)#advertise-definition
```

### Verification

This following show output displays the route-type as 'Extern' for the external routes.

```
Router#show ospf routes flex-algo 240 route-type external detail
Route Table of ospf-1 with router ID 192.168.0.2 (VRF default)

  Algorithm 240

  Route entry for 192.168.4.3/32, Metric 220, SID 536, Label 16536
   Priority : Medium
```

```
     Route type : Extern Type 1
Last updated : Apr 25 14:30:12.718
     Flags: Inuse

     Prefix Contrib Algo 240 SID 536
      From 192.168.0.4 Route-type 5
      Total Metric : 220 Base metric 20 FAPM 20
      Contrib Flags : Inuse, Reachable
      SID Flags : PHP off, Index, Global, Valid

      Path: 10.1.1.3, from 192.168.0.4, via GigabitEthernet0/2/0/2
       Out Label  : 16536
       Weight     : 0
       Area       : 0

      Path: 10.1.2.3, from 192.168.0.4, via GigabitEthernet0/2/0/3
       Out Label  : 16536
       Weight     : 0
       Area       : 0

      Path: 10.2.1.5, from 192.168.0.4, via GigabitEthernet0/2/0/4
       Out Label  : 16536
       Weight     : 0
       Area       : 0

  Route entry for 192.168.4.5/32, Metric 120, SID 556, Label 16556
   Priority : Medium

     Route type : Extern Type 1
     Last updated : Apr 25 14:30:12.724
     Flags: Inuse

     Prefix Contrib Algo 240 SID 556
      From 192.168.0.3 Route-type 5
      Total Metric : 120 Base metric 1 FAPM 20
      Contrib Flags : Inuse, Reachable
      SID Flags : PHP off, Index, Global, Valid

      Path: 10.1.1.3, from 192.168.0.3, via GigabitEthernet0/2/0/2
       Out Label  : 16556
       Weight     : 0
       Area       : 0

      Path: 10.1.2.3, from 192.168.0.3, via GigabitEthernet0/2/0/3
       Out Label  : 16556
       Weight     : 0
       Area       : 0
```

The following show output displays label information for flexible algorithm and its corresponding metric as added in RIB:

```
RP/0/RP0/CPU0:ios# show route 192.168.0.2/32 detail
Wed Apr  6 16:24:46.021 IST

Routing entry for 192.168.0.2/32
  Known via "ospf 1", distance 110, metric 2, labeled SR, type intra area
  Installed Apr  6 15:51:57.973 for 00:32:48
  Routing Descriptor Blocks
    10.10.10.2, from 192.168.0.2, via GigabitEthernet0/2/0/0, Protected
      Route metric is 2
      Label: 0x3 (3)
      Tunnel ID: None
      Binding Label: None
```

```
    Extended communities count: 0
    Path id:1       Path ref count:0
    NHID:0x1(Ref:1)
    Backup path id:65
    OSPF area: 1
  10.11.11.2, from 192.168.0.2, via GigabitEthernet0/2/0/1, Backup (Local-LFA)
    Route metric is 6
    Label: 0x3 (3)
    Tunnel ID: None
    Binding Label: None
    Extended communities count: 0
    Path id:65              Path ref count:1
    NHID:0x2(Ref:1)
    OSPF area:
Route version is 0x12 (18)
Local Label: 0x3ee6 (16102)
Local Label Algo Set (ID, Label, Metric): (1, 16202, 0),(128, 17282, 2)
IP Precedence: Not Set
QoS Group ID: Not Set
Flow-tag: Not Set
Fwd-class: Not Set
Route Priority: RIB_PRIORITY_NON_RECURSIVE_MEDIUM (7) SVD Type RIB_SVD_TYPE_LOCAL
Download Priority 1, Download Version 38
No advertising protos.
```

# Flexible Algorithm Prefix Metric

*Table 48: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Prefix Metric support for OSPF Flexible Algorithm | Release 7.5.1 | This feature extends the current OSPF Flexible Algorithm functionality to introduce a Flexible Algorithm-specific prefix-metric in the OSPF prefix advertisement. The prefix-metric provides a way to compute the best end-to-end Flexible Algorithm optimized paths across multiple areas or domains. |

A limitation of the existing Flexible Algorithm functionality in IS-IS and OSPF is the inability to compute the best path to a prefix in a remote area or remote IGP domain. Prefixes are advertised between IS-IS areas, OSPF processes, or between protocol domains, but the existing prefix metric does not reflect any of the constraints used for Flexible Algorithm path. Although the best Flexible Algorithm path can be computed to the inter-area or redistributed prefix inside the area, the path may not represent the overall best path through multiple areas or IGP domains.

The Flexible Algorithm Prefix Metric feature introduces a Flexible Algorithm-specific prefix-metric in the IS-IS and OSPF prefix advertisement. The prefix-metric provides a way to compute the best end-to-end Flexible Algorithm optimized paths across multiple areas or domains.

**Note** The Flexible Algorithm definition must be consistent between domains or areas. Refer to section 8 and section 9 in IETF draft https://datatracker.ietf.org/doc/draft-ietf-lsr-flex-algo/.

# Configuring Flexible Algorithm

*Table 49: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| TE Metric Support for IS-IS Flex Algo | Release 7.4.1 | Flexible Algorithm allows for user-defined algorithms where the IGP computes paths based on a user-defined combination of metric type (path optimization objective) and constraint.<br>This feature adds support for TE metric as a metric type for IS-IS Flexible Algorithm. This allows the TE metric, along with IGP and delay metrics, to be used when running shortest path computations. |

The following IS-IS and OSPF configuration sub-mode is used to configure Flexible Algorithm:

```
router isis instance flex-algo algo
```

```
router ospf process flex-algo algo
```

*algo*—value from 128 to 255

### Configuring Flexible Algorithm Definitions

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

- `router isis instance flex-algo algo metric-type {delay | te}`

  `router ospf process flex-algo algo metric-type {delay | te-metric}`

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

**Note** See Flexible Algorithm Link Attribute Advertisement Behavior, on page 418 for TE metric behaviors.

- `router isis instance flex-algo algo affinity { include-any | include-all | exclude-any} name1, name2, …`

  `router ospf process flex-algo algo affinity { include-any | include-all | exclude-any} name1, name2, …`

*name*—name of the affinity map

- **router isis** *instance* **flex-algo** *algo* **priority** *priority value*

  **router ospf** *process* **flex-algo** *algo* **priority** *priority value*

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

- IS-IS

  **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.

OSPF

**metric-type** {**delay** | **te-metric**}

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

- **affinity** {**include-any** | **include-all** | **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 to include the Flexible Algorithm prefix metric in the advertised Flexible Algorithm definition in IS-IS and OSPF :

**router isis** *instance* **flex-algo** *algo* **prefix-metric**

**router ospf** *process* **flex-algo** *algo* **prefix-metric**

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

**router isis** *instance* **flex-algo** *algo* **advertise-definition**

### Configuring Affinity

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.

**router isis** *instance* **flex-algo** *algo* **affinity-map** *name* **bit-position** *bit number*

**router ospf** *process* **flex-algo** *algo* **affinity-map** *name* **bit-position** *bit number*

*name*—name of the affinity-map

### Configuring Prefix-SID Advertisement

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

```
router isis instance interface type interface-path-id address-family {ipv4 | ipv6} [unicast]
 prefix-sid [strict-spf | algorithm algorithm-number]  [index | absolute] sid value

router ospf process area area interface Loopback interface-instance prefix-sid [strict-spf
 | algorithm algorithm-number]  [index | absolute] sid value
```

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

# Flexible Algorithm Link Attribute Advertisement Behavior

***Table 50: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Advertisement of Link Attributes for IS-IS Flexible Algorithm | Release 7.4.1 | Link attribute advertisements used during Flexible Algorithm path calculation must use the Application-Specific Link Attribute (ASLA) advertisements, as defined in IETF draft draft-ietf-lsr-flex-algo.<br>This feature introduces support for ASLA advertisements during IS-IS Flexible Algorithm path calculation. |

The following tables explain the behaviors for advertising (transmitting) and processing (receiving) Flexible Algorithm link attributes.

***Table 51: OSPF***

| Link Attribute | Transmit | Receive |
|---|---|---|
| Link Delay Metric | IOS XR OSPF Flex Algo implementation advertises the link delay metric value using the OSPF ASLA sub-TLV with the F-bit set. | IOS XR OSPF only uses the link delay metric advertised in the ASLA sub-TLV for Flex Algo.<br>ASLA sub-TLV is supported with non-zero-length or with zero-length Application Identifier Bit Masks. |

| Link Attribute | Transmit | Receive |
|---|---|---|
| Link TE Metric | IOS XR OSPF Flex Algo implementation advertises the link TE metric value using the OSPF ASLA sub-TLV with the F-bit set.<br>The link TE metric values advertised are configured under SR-TE. | IOS XR OSPF only uses the TE metric advertised in the ASLA sub-TLV for Flex Algo.<br>ASLA sub-TLV is supported with non-zero-length or with zero-length Application Identifier Bit Masks. |
| Link Admin Group/Extended Admin Group | IOS XR OSPF Flex Algo implementation advertises the link admin group value using both link admin group (AG) and link extended admin group (EAG) encoding using the OSPF ASLA sub-TLV with the F-bit set.<br>The link admin group values advertised can be configured directly under the IGP and are therefore FA-specific. Otherwise, they will be derived from the link admin group values configured under SR-TE. | IOS XR OSPF only uses the AG/EAG (either one or both) advertised in the ASLA sub-TLV for Flex Algo.<br>ASLA sub-TLV is supported with non-zero-length or with zero-length Application Identifier Bit Masks. |

***Table 52: IS-IS***

| | | |
|---|---|---|
| Link Delay Metric | IOS XR IS-IS Flex Algo implementation advertises the link delay metric value using the IS-IS Extended Reachability TLV only. | IOS XR IS-IS Flex Algo implementation processes the link delay metric value received in the IS-IS Extended Reachability TLV only. |
| Link Extended Admin Group | IOS XR IS-IS Flex Algo implementation advertises the affinity value using the link extended admin group TLV using the IS-IS ASLA. | IOS XR IS-IS Flex Algo implementation processes the affinity value received in the link extended admin group TLV in the IS-IS ASLA. |
| Link SRLG | IOS XR IS-IS LFA implementation advertises the link SRLG value in the IS-IS ASLA. | IOS XR IS-IS LFA implementation processes the link SRLG value received in the IS-IS ASLA. |

*Table 53: IS-IS*

| Link Attribute | Transmit | Receive |
|---|---|---|
| Link Delay Metric | IOS XR IS-IS Flex Algo implementation advertises the link delay metric value using **both** the IS-IS Extended Reachability TLV and the IS-IS ASLA. | By default, IOS XR IS-IS Flex Algo implementation prefers the link delay metric value received in the IS-IS ASLA. Otherwise, it will use link delay metric value received in the IS-IS Extended Reachability TLV.<br>ASLA sub-TLV is supported with non-zero-length or with zero-length Application Identifier Bit Masks.<br>If the incoming ASLA includes the L-Flag, implementation derives the link delay metric value from the IS-IS Extended Reachability TLV.<br>You can configure the IOS XR IS-IS Flex Algo implementation to strictly use the link delay metric value received in the IS-IS ASLA. See Strict IS-IS ASLA Link Attribute, on page 421. |
| Link TE Metric | IOS XR IS-IS Flex Algo implementation advertises the link TE metric value using the IS-IS ASLA.<br>The link TE metric values advertised can be configured directly under the IGP and are therefore FA-specific. Otherwise, they will be derived from the link TE metric values configured under SR-TE.<br>See Flexible Algorithm-Specific TE Metric, on page 421. | IOS XR IS-IS Flex Algo implementation processes the link TE metric value received in the IS-IS ASLA.<br>ASLA sub-TLV is supported with non-zero-length or with zero-length Application Identifier Bit Masks.<br>If incoming ASLA includes the L-Flag, implementation derives the link TE metric value from the IS-IS Extended Reachability TLV. |
| Link Admin Group/Extended Admin Group | IOS XR IS-IS Flex Algo implementation advertises the affinity value as both the link admin group (AG) TLV and the link extended admin group (EAG) TLV using the IS-IS ASLA when its value falls within the first 32 bits. Otherwise, the affinity value is advertised only as link EAG TLV using the IS-IS ASLA.<br>The admin group values advertised are configured directly under the IGP and are therefore FA-specific. | IOS XR IS-IS Flex Algo implementation processes the affinity value received as either the link admin group TLV or link extended admin group TLV in the IS-IS ASLA.<br>ASLA sub-TLV is supported with non-zero-length or with zero-length Application Identifier Bit Masks.<br>If incoming ASLA includes the L-Flag, implementation derives the affinity value from the IS-IS Extended Reachability TLV. |

| Link Attribute | Transmit | Receive |
|---|---|---|
| Link SRLG | IOS XR IS-IS LFA implementation advertises the link SRLG value in the IS-IS ASLA. | IOS XR IS-IS LFA implementation processes the link SRLG value received in the IS-IS ASLA.<br>If incoming ASLA includes the L-Flag, implementation derives the link SRLG value from the IS-IS Extended Reachability TLV. |

# Strict IS-IS ASLA Link Attribute

Use the following command to configure the IOS XR IS-IS Flex Algo implementation to strictly use the link delay metric value received in the IS-IS ASLA:

```
router isis instance-id receive application flex-algo delay app-only
```

# Flexible Algorithm-Specific TE Metric

Use the following command to configure the Flexible Algorithm-specific TE metric value under IS-IS, where *metric_value* is from 1 to 16777214:

- **router isis** *instance* **interface** *type interface-path-id* **address-family** { **ipv4** | **ipv6**} [**unicast**] **te-metric flex-algo** *metric_value* [**level** {**1** | **2**}]

The following example shows how to configure the IS-IS Flexible Algorithm-specific TE metric value to 50:

```
Router(config)# router isis 1
Router(config-isis)# interface HundredGigE 0/0/0/2
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# te-metric flex-algo 50
```

Use the following command to configure the Flexible Algorithm-specific TE metric value under OSPF, where *metric_value* is from 1 to 2147483647:

- **router ospf** *process-name* **area** *area* **interface** *type interface-path-id* **te-metric flex-algo** *metric_value*

The following example shows how to configure the OSPF Flexible Algorithm-specific TE metric value to 50:

```
Router(config)# router ospf 1
Router(config-ospf)# area 0
Router(config-ospf-ar)# interface HundredGigE 0/0/0/2
Router(config-ospf-ar-if# te-metric flex-algo 50
```

# Flexible Algorithm with Exclude SRLG Constraint

*Table 54: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Flexible Algorithm to Exclude SRLGs for OSPF | Release 7.5.2 | You can now configure the flexible algorithm to exclude any link belonging to the Shared Risk Link Groups (SRLGs) from the path computation for OSPF. The ability to exclude the at-risk links ensures that the rest of the links in the network remain unaffected. |
| IS-IS Flexible Algorithm: Exclude-SRLG Constraint | Release 7.5.1 | This feature allows the Flexible Algorithm definition to specify Shared Risk Link Groups (SRLGs) that the operator wants to exclude during the Flex-Algorithm path computation. The ability to exclude the at-risk links ensures that the rest of the links in the network remain unaffected.<br>This allows the setup of disjoint paths between two or more Flex Algos by leveraging deployed SRLG configurations. |

This feature allows the Flexible Algorithm definition to specify Shared Risk Link Groups (SRLGs) that the operator wants to exclude during the Flex-Algorithm path computation. A set of links that share a resource whose failure can affect all links in the set constitute a SRLG. An SRLG provides an indication of which links in the network might be at risk from the same failure.

This allows the setup of disjoint paths between two or more Flex Algos by leveraging deployed SRLG configurations. For example, multiple Flex Algos could be defined by excluding all SRLGs except one. Each FA will prune the links belonging to the excluded SRLGs from its topology on which it computes its paths.

This provides a new alternative to creating disjoint paths with FA, in addition to leveraging FA with link admin group (affinity) constraints.

The Flexible Algorithm definition (FAD) can advertise SRLGs that you want to exclude during the Flexible Algorithm path computation. The IS-IS Flexible Algorithm Exclude SRLG Sub-TLV (FAESRLG) is used to advertise the exclude rule that is used during the Flexible Algorithm path calculation, as specified in IETF draft https://datatracker.ietf.org/doc/draft-ietf-lsr-flex-algo/

The Flexible Algorithm path computation checks if an “exclude SRLG” rule is part of the FAD. If an “exclude SRLG” rule exists, it then checks if the link is part of an SRLG that is also part of the “exclude SRLG” rule. If the link is part of an excluded SRLG, the link is pruned from the path computation.

The figure below shows a topology configured with the following flex algos:

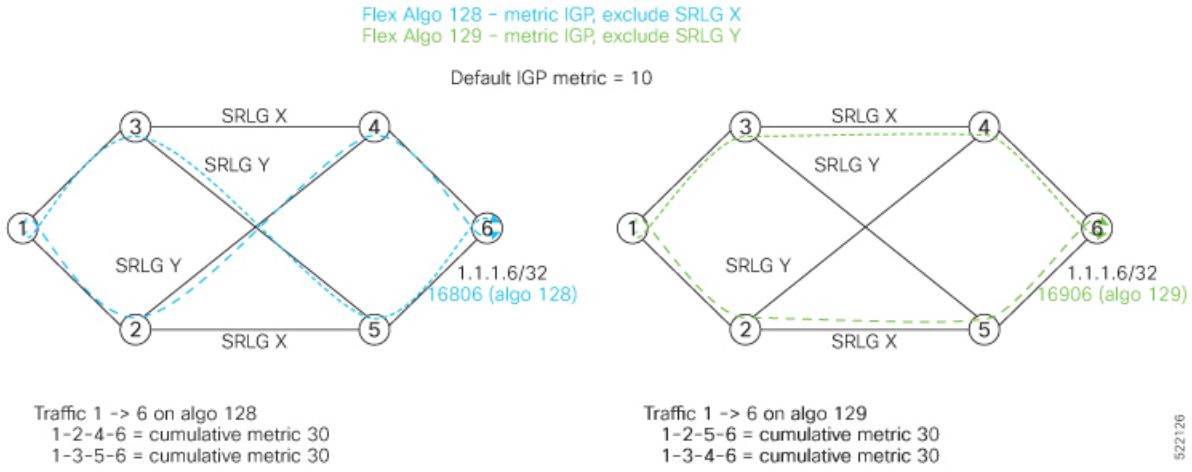
- Flex algo 128: metric IGP and exclude SRLG X constraint

- Flex algo 129: metric IGP and exclude SRLG Y constraint

The horizontal links between nodes 3 and 4 and between 2 and 5 are part of SRLG group X. The diagonal links between nodes 3 and 5 and between 2 and 4 are part of SRLG group Y. As a result, traffic from node 1 to node 6's FA 128 prefix SID (16806) avoids interfaces part of SRLG X. While traffic from node 1 to node 6's FA 129 prefix SID (16906) avoids interfaces part of SRLG Y.

**Note** See Constraints, on page 205 section in the *Configure SR-TE Policies* chapter for information about configuring SR policies with Flex-Algo constraints.

***Figure 34: Flex Algo with Exclude SRLG Constraint***




### Configuration

Use the **router isis** *instance* **address-family ipv4 unicast advertise application flex-algo link-attributes srlg** command to enable the Flexible Algorithm ASLA-specific advertisement of SRLGs.

Use the **router isis** *instance* **flex-algo** *algo* **srlg exclude-any** *srlg-name . . . srlg-name* command to configure the SRLG constraint which is advertised in the Flexible Algorithm definition (FAD) if the FAD advertisement is enabled under the flex-algo sub-mode. You can specify up to 32 SRLG names.

The SRLG configuration (value and port mapping) is performed under the global SRLG sub-mode. Refer to MPLS Traffic Engineering Shared Risk Link Groups for more information.

### Example

The following example shows how to enable the Flexible Algorithm ASLA-specific advertisement of SRLGs and to exclude SRLG groups from Flexible Algorithm path computation:

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface HunGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name groupX
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 groupX
RP/0/RP0/CPU0:router(config-srlg-if)# exit

RP/0/RP0/CPU0:router(config-srlg)# interface HunGigE0/0/1/0
```

```
RP/0/RP0/CPU0:router(config-srlg-if)# name groupY
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/1/1
RP/0/RP0/CPU0:router(config-srlg-if)# name groupY
RP/0/RP0/CPU0:router(config-srlg-if)# exit

RP/0/RP0/CPU0:router(config-srlg)# name groupX value 100
RP/0/RP0/CPU0:router(config-srlg)# name groupY value 200
RP/0/RP0/CPU0:router(config-srlg)# 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-af)# advertise application flex-algo link-attributes srlg
RP/0/RP0/CPU0:router(config-isis-af)# exit
RP/0/RP0/CPU0:router(config-isis)# flex-algo 128
RP/0/RP0/CPU0:router(config-isis-flex-algo)# advertise-definition
RP/0/RP0/CPU0:router(config-isis-flex-algo)# srlg exclude-any groupX
RP/0/RP0/CPU0:router(config-isis-flex-algo)# exit
RP/0/RP0/CPU0:router(config-isis)# flex-algo 129
RP/0/RP0/CPU0:router(config-isis-flex-algo)# advertise-definition
RP/0/RP0/CPU0:router(config-isis-flex-algo)# srlg exclude-any groupY
RP/0/RP0/CPU0:router(config-isis-flex-algo)# commit
RP/0/RP0/CPU0:router(config-isis-flex-algo)# exit
RP/0/RP0/CPU0:router(config-isis)#
```

The following example shows how to enable the Flexible Algorithm ASLA-specific advertisement of SRLGs and to exclude SRLG groups from Flexible Algorithm path computation for OSPF:

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface HunGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name groupX
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 groupX
RP/0/RP0/CPU0:router(config-srlg-if)# exit

RP/0/RP0/CPU0:router(config-srlg)# interface HunGigE0/0/1/0
RP/0/RP0/CPU0:router(config-srlg-if)# name groupY
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/1/1
RP/0/RP0/CPU0:router(config-srlg-if)# name groupY
RP/0/RP0/CPU0:router(config-srlg-if)# exit

RP/0/RP0/CPU0:router(config-srlg)# name groupX value 100
RP/0/RP0/CPU0:router(config-srlg)# name groupY value 200
RP/0/RP0/CPU0:router(config-srlg)# exit

RP/0/0/CPU0:r1(config)#router ospf 1
RP/0/0/CPU0:r1(config-ospf)#flex-algo 128
RP/0/0/CPU0:r1(config-ospf-flex-algo)#srlg exclude-any
RP/0/0/CPU0:r(config-ospf-flex-algo-srlg-exclude-any)#groupX
RP/0/0/CPU0:r(config-ospf-flex-algo-srlg-exclude-any)#groupY
RP/0/0/CPU0:r(config-ospf-flex-algo-srlg-exclude-any)#commit
```

### Verification

The following example shows how to verify the number of SRLGs excluded for OSPF:

```
RP/0/RP0/CPU0:router# show ospf topology summary
```

```
 Process ospf-1
 Instance default
  Router ID       : 192.168.0.1
  Number of Areas : 1
  Number of Algos : 1
  Max Path count  : 16
  Route count     : 10
  SR Global Block : 16000 - 23999

Area 0
    Number of Nodes : 6
     Algo 128
      FAD Advertising Router : 192.168.0.1
      FAD Area ID : 0
       Algo Type   : 0
       Metric Type : 0
          Number of Exlclude SRLGs  : (2)
          [1]:  100        [2]:  200
       FAPM supported : No
```

# 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
  affinity exclude-any red
  affinity include-any blue
 !
 flex-algo 129
  affinity exclude-any 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
!
```

# Example: Configuring OSPF Flexible Algorithm

```
router ospf 1
 flex-algo 130
  priority 200
  affinity exclude-any
   red
   blue
  !
  metric-type delay
 !
 flex-algo 140
  affinity include-all
   green
  !
  affinity include-any
   red
  !
 !

  interface Loopback0
   prefix-sid index 10
   prefix-sid strict-spf index 40
   prefix-sid algorithm 128 absolute 16128
   prefix-sid algorithm 129 index 129
   prefix-sid algorithm 200 index 20
   prefix-sid algorithm 210 index 30
  !
 !

  interface GigabitEthernet0/0/0/0
   flex-algo affinity
    color red
    color blue
   !
  !

 affinity-map
  color red bit-position 10
  color blue bit-position 11
 !
```

# 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
  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 10.1.1.1
  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
```

# Delay Normalization

***Table 55: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| SR-TE Delay Normalization for OSPF | Release 7.3.1 | This feature extends the current Delay Normalization feature to support OSPF. |

Performance measurement (PM) measures various link characteristics like packet loss and delay. Such characteristics can be used by IS-IS as a metric for Flexible Algorithm computation. Low latency routing using dynamic delay measurement is one of the primary use cases for Flexible Algorithm technology.

Delay is measured in microseconds. If delay values are taken as measured and used as link metrics during the IS-IS topology computation, some valid ECMP paths might be unused because of the negligible difference in the link delay.

The Delay Normalization feature computes a normalized delay value and uses the normalized value instead. This value is advertised and used as a metric during the Flexible Algorithm computation.

The normalization is performed when the delay is received from the delay measurement component. When the next value is received, it is normalized and compared to the previous saved normalized value. If the values are different, then the LSP generation is triggered.

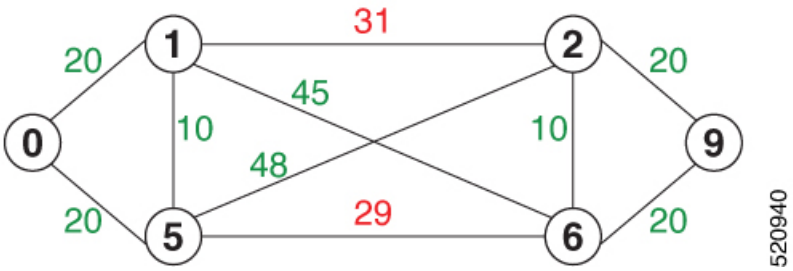
The following formula is used to calculate the normalized value:

- **Dm** – measured Delay
- **Int** – configured normalized Interval
- **Off** – configured normalized Offset (must be less than the normalized interval Int)
- **Dn** – normalized Delay
- **a** = Dm / Int (rounded down)
- **b** = a * Int + Off

If the measured delay (Dm) is less than or equal to **b**, then the normalized delay (Dn) is equal to **b**. Otherwise, Dn is **b** + **Int**.

### Example

The following example shows a low-latency service. The intent is to avoid high-latency links (1-6, 5-2). Links 1-2 and 5-6 are both low-latency links. The measured latency is not equal, but the difference is insignificant.




We can normalize the measured latency before it is advertised and used by IS-IS. Consider a scenario with the following:

- Interval = 10
- Offset = 3

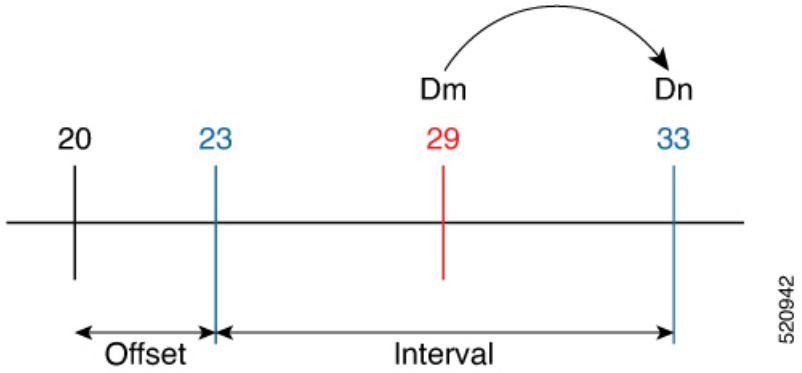
The measured delays will be normalized as follows:

- **Dm** = 29

  **a** = 29 / 10 = 2 (2.9, rounded down to 2)

  **b** = 2 * 10 + 3 = 23

  In this case, **Dm** (29) is greater than **b** (23); so **Dn** is equal to **b**+**I** (23 + 10) = **33**

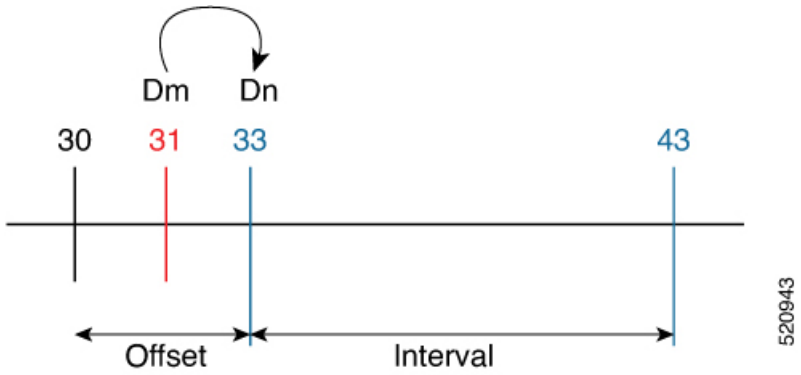



- **Dm** = 31

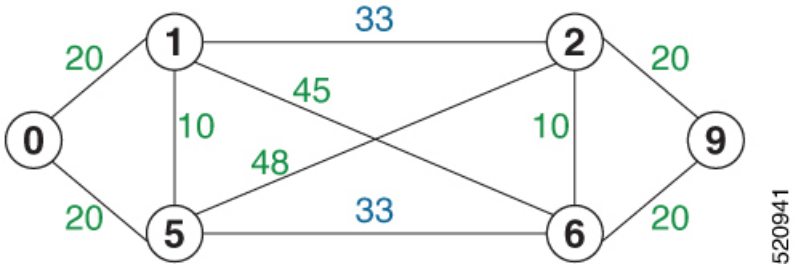
  **a** = 31 / 10 = 3 (3.1, rounded down to 3)

  **b** = 3 * 10 + 3 = 33

  In this case, **Dm** (31) is less than **b** (33); so **Dn** is **b** = **33**




The link delay between 1-2 and 5-6 is normalized to 33.




### Configuration

Delay normalization is disabled by default. To enable and configure delay normalization, use the **delay normalize interval** *interval* [**offset** *offset*] command.

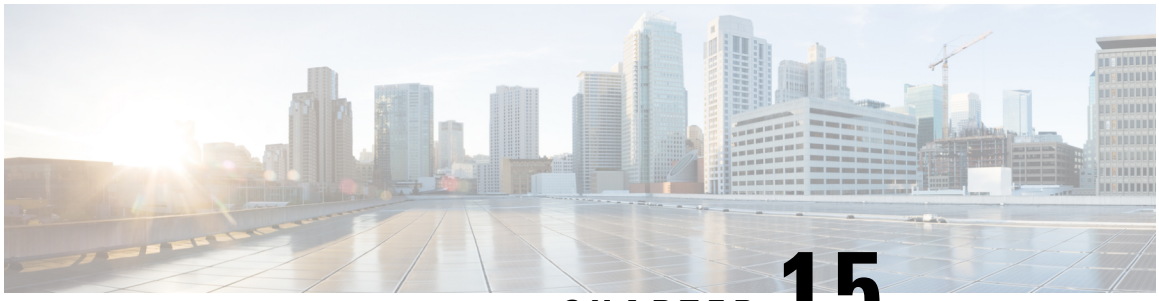
- *interval* – The value of the normalize interval in microseconds.
- *offset* – The value of the normalized offset in microseconds. This value must be smaller than the value of normalized interval.

### IS-IS Configuration

```
router isis 1
 interface GigEth 0/0/0/0
  delay normalize interval 10 offset 3
  address-family ipv4 unicast
   metric 77
```

### OSPF Configuration

```
router ospf 1
 area 0
  interface GigabitEthernet0/0/0/0
   delay normalize interval 10 offset 3
  !
 !
!
```



CHAPTER 15

# 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 SIDs, IGP prefix SIDs, and Nil-FEC (forwarding equivalence classes) LSP Ping and Traceroute functionality.



# MPLS Ping and Traceroute for BGP and IGP Prefix-SID

***Table 56: Feature History Table***

| Feature Name | Release | Description |
|---|---|---|
| MPLS Ping and Traceroute for BGP and IGP Prefix-SID | Release 7.6.1 | The MPLS LSP Ping feature is used to check the connectivity between ingress Label Switch Routers (LSRs) and egress LSRs along an LSP. |

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.

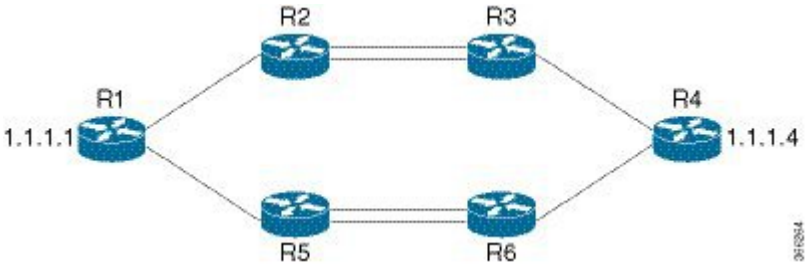
The MPLS LSP Traceroute feature is used to isolate the failure point of an LSP. It is used for hop-by-hop fault localization and path tracing. The MPLS LSP Traceroute feature relies on the expiration of the Time to Live (TTL) value of the packet that carries the echo request. When the MPLS echo request 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:




### MPLS Ping for Prefix-SID

```
RP/0/RP0/CPU0:router-arizona# ping mpls ipv4 10.1.1.4/32
Thu Dec 17 01:01:42.301 PST

Sending 5, 100-byte MPLS Echos to 10.1.1.4,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
```

```
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/2/3 ms
```

### MPLS Traceroute for Prefix-SID

```
RP/0/RP0/CPU0:router-arizona# traceroute mpls ipv4 10.1.1.4/32
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/RP0/CPU0:router-arizona# traceroute mpls multipath ipv4 10.1.1.4/32
Thu Dec 17 14:55:46.549 PST

Starting LSP Path Discovery for 10.1.1.4/32

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

***Table 57: Feature History Table***

| Feature Name | Release | Description |
|---|---|---|
| MPLS LSP Ping and Traceroute Nil FEC Target | Release 7.6.1 | This feature allows operators to provide the ability to freely test any label stack by allowing them to specify the following:<br>• label stack<br>• outgoing interface<br>• nexthop address |

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 58: LSP Ping and Traceroute Nil FEC Commands***

| Command Syntax |
|---|
| **ping mpls nil-fec labels** {*label*[**,***label*]} [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]] |
| **traceroute mpls nil-fec labels** {*label*[**,***label*]} [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]] |

# Examples: LSP Ping 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:            GigabitEthernet0/0/0/1   GigabitEthernet0/0/0/1
Interface IP address:         10.1.1.3              10.1.1.4


RP/0/RP0/CPU0:router-utah# show mpls forwarding

Tue Jul  5 13:44:31.999 EDT
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
16004  Pop         No ID              Gi0/0/0/1    10.1.1.4        1392
       Pop         No ID              Gi0/0/0/2    10.1.2.2        0
16005  16005       No ID              Gi0/0/0/0    10.1.1.4        0
       16005       No ID              Gi0/0/0/1    10.1.2.2        0
16007  16007       No ID              Gi0/0/0/0    10.1.1.4        4752
       16007       No ID              Gi0/0/0/1    10.1.2.2        0
24000  Pop         SR Adj (idx 0)     Gi0/0/0/0    10.1.1.4        0
24001  Pop         SR Adj (idx 2)     Gi0/0/0/0    10.1.1.4        0
24002  Pop         SR Adj (idx 0)     Gi0/0/0/1    10.1.2.2        0
24003  Pop         SR Adj (idx 2)     Gi0/0/0/1    10.1.2.2        0
24004  Pop         No ID              tt10         point2point     0
24005  Pop         No ID              tt11         point2point     0
24006  Pop         No ID              tt12         point2point     0
24007  Pop         No ID              tt13         point2point     0
24008  Pop         No ID              tt30         point2point     0
```

### Ping Nil FEC Target

```
RP/0/RP0/CPU0:router-arizona# ping mpls nil-fec labels 16005,16007 output interface
GigabitEthernet 0/0/0/1 nexthop 10.1.1.4 repeat 1
Sending 1, 72-byte MPLS Echos with Nil FEC labels 16005,16007,
     timeout is 2 seconds, send interval is 0 msec:

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

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

### Traceroute Nil FEC Target

```
RP/0/RP0/CPU0:router-arizona# traceroute mpls nil-fec labels 16005,16007 output interface
GigabitEthernet 0/0/0/1 nexthop 10.1.1.4
Tracing MPLS Label Switched Path with Nil FEC labels 16005,16007, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
```

```
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'd' - see DDMAP for return code,
  'X' - unknown return code, 'x' - return code 0

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

# Segment Routing Ping and Traceroute

***Table 59: Feature History Table***

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Segment Routing Ping and Traceroute | Release 7.6.1 | Segment routing ping and traceroute features extendsthe MPLS LSP ping and traceroute functionality to perform the connectivity verification on the segment routing control plane. |
| SR OAM for SR Policy (Policy Name / Binding SID / Custom label stack) | Release 7.3.1 | This feature extends SR OAM ping and traceroute function for an SR policy (or binding SID)-LSP end-point combination.<br>This addresses the limitations of the Nil-FEC LSP Ping and Traceroute function which cannot perform a ping operation to a segment list that is not associated with an installed SR policy. Also, it cannot validate egress device-specific SR policies. |

## Segment Routing Ping

The MPLS LSP ping feature is used to check the connectivity between ingress and egress of LSP. MPLS LSP ping uses MPLS echo request and reply messages, similar to Internet Control Message Protocol (ICMP) echo request and reply messages, to validate an LSP. Segment routing ping is an extension of the MPLS LSP ping to perform the connectivity verification on the segment routing control plane.

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

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

protocol is unknown or might change during the path of the echo request. If you need to specify the target FEC, you can select the FEC type as OSPF, IS-IS, or BGP. This ensures that only devices that are running segment routing control plane, and can therefore understand the segment routing IGP FEC, respond to the echo request.

### Configuration Examples

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

```
RP/0/RP0/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/RP0/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/RP0/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/RP0/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/RP0/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
```

### Ping for SR Policy

You can perform the ping operation for an SR policy (or binding SID), and LSP end-point combination. Use the **ping** command's **policy name lsp-end-point** and **policy binding-sid lsp-end-point** options to perform this task. You can instantiate the policy through the CLI, Netconf, PCEP or BGP-TE process.

IPv6 policies are not supported for SR OAM function.

**Note** As a prerequisite, you must enable the MPLS OAM function.

```
Router(config)# mpls oam
Router(config)# commit

Router# ping sr-mpls policy name srte_c_4_ep_10.0.0.1 lsp-end-point 209.165.201.1
Router# ping sr-mpls policy binding-sid 1000 lsp-end-point 209.165.201.1
```

# Segment Routing Traceroute

The MPLS LSP traceroute is used to isolate the failure point of an LSP. It is used for hop-by-hop fault localization and path tracing. The MPLS LSP traceroute feature relies on the expiration of the Time to Live (TTL) value of the packet that carries the echo request. When the MPLS echo request message hits a transit node, it checks the TTL value and if it is expired, the packet is passed to the control plane, else the message is forwarded. If the echo message is passed to the control plane, a reply message is generated based on the contents of the request message. Segment routing traceroute feature extends the MPLS LSP traceroute functionality to segment routing networks.

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

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

### Configuration Examples

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

```
RP/0/RP0/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

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 3 ms


RP/0/RP0/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

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/RP0/CPU0: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/RP0/CPU0: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/RP0/CPU0: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/RP0/CPU0:router# traceroute sr-mpls multipath 10.1.1.2/32

Starting LSP Path Discovery for 10.1.1.2/32
```

```
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.

!
Path 0 found,
 output interface GigabitEthernet0/0/0/2 nexthop 10.13.13.2
source 10.13.13.1 destination 127.0.0.0
!
Path 1 found,
 output interface Bundle-Ether1 nexthop 10.12.12.2
source 10.12.12.1 destination 127.0.0.0

Paths (found/broken/unexplored) (2/0/0)
Echo Request (sent/fail) (2/0)
Echo Reply (received/timeout) (2/0)
Total Time Elapsed 14 ms
```

### Traceroute for SR Policy

You can perform the traceroute operation for an SR policy (or binding SID), and LSP end-point combination. Use the **traceroute** command's **policy name lsp-end-point** and **policy binding-sid lsp-end-point** options to perform this task. You can instantiate the policy through the CLI, Netconf, PCEP or BGP-TE process.

IPv6 policies are not supported for SR OAM function.

**Note** As a prerequisite, you must enable the MPLS OAM function.

```
Router(config)# mpls oam
Router(config)# commit

Router# traceroute sr-mpls policy name srte_c_4_ep_10.0.0.1 lsp-end-point 209.165.201.1
Router# traceroute sr-mpls policy binding-sid 1000 lsp-end-point 209.165.201.1
```

# Segment Routing Ping and Traceroute for Flexible Algorithm

*Table 60: Feature History Table*

| Feature Name | Release Information | Feature Description |
|---|---|---|
| Segment Routing Ping and Traceroute for Flexible Algorithm | Release 7.6.1 | Flexible Algorithm validation method is based on segment identifier (SID) label and label switched path (LSP) destination, instead of being based on IP address. |

Flexible Algorithm validation method is based on segment identifier (SID) label and label switched path (LSP) destination, instead of being based on IP address. The assigner is validated against the topology prefix information provided by SR-PCE database. If the assigner is valid, then the label given is also validated against

the SR-PCE database. On the egress side, the destination label is contained in a new SR Label sub-TLV. This label is verified against a SID list provided by SR-PCE.

**Note** Observe the following guidelines and restrictions:

- All routers within an area must share the same Flexible Algorithm definition for a Flexible Algorithm to be valid.
- All routers within the domain must be configured with the same SRGB range of values.
- BGP-LS must be enabled.
- Only prefix SIDs and Flexible Algorithm SIDs are supported.
- Only single label stack is supported.

# Segment Routing Ping for Flexible Algorithm

```
Router# ping sr-mpls labels 16131 lsp-end-point 10.1.1.5
Fri Dec 13 19:26:29.517 IST

Sending 5, 100-byte MPLS Echos with SR Label FEC with lsp end point 10.1.1.5, SID Label(s)
 [16131],
      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 = 4/4/6 ms
```

# Segment Routing Traceroute for Flexible Algorithm

```
Router# traceroute sr-mpls labels 16130 lsp-end-point 10.1.1.5
Fri Dec 13 19:26:59.368 IST

Tracing MPLS Label Switched Path to SR Label FEC with lsp end point 10.1.1.5, SID Label(s)
 [16130], 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 13.13.13.1 MRU 1500 [Labels: 16130 Exp: 0]
L 1 13.13.13.3 MRU 1500 [Labels: 16130 Exp: 0] 5 ms
L 2 16.16.16.4 MRU 1500 [Labels: implicit-null Exp: 0] 4 ms
! 3 18.18.18.5 4 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::33:33:33:33 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
 1  2001::22:22:22:22[IP tunnel: DA=cafe:0:0:a4:1:::: SRH =(2001::33:33:33:33   ,SL=1)] 2
msec
 2  2001::2:2:2:2[IP tunnel: DA=cafe:0:0:a4:1:::: SRH =(2001::33:33:33:33   ,SL=1)] 2 msec
 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
```