



Segment Routing Configuration Guide for Cisco NCS 6000 Series Routers, IOS XR Release 7.4.x

First Published: 2021-07-30

Last Modified: 2022-02-28

Americas Headquarters

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



CONTENTS

PREFACE

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

CHAPTER 1

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

CHAPTER 2

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

CHAPTER 3

Configure Segment Routing over IPv6 (SRv6)	7
Segment Routing over IPv6 Overview	7
SRv6 Micro-Segment (uSID)	12
SRv6 uSID Terminology	13
SRv6 uSID Carrier Format	13
SRv6 uSID Allocation Within a uSID Block	14
SRv6 Endpoint Behaviors Associated with uSID	14
SRv6 uSID in Action - Example	15
Usage Guidelines and Limitations	20
Configuring SRv6	21
Configuring SRv6 under IS-IS	23

CHAPTER 4

Configure Segment Routing Global Block	25
---	-----------

About the Segment Routing Global Block 25
 Setup a Non-Default Segment Routing Global Block Range 27

CHAPTER 5

Configure Segment Routing for IS-IS Protocol 29
 Enabling Segment Routing for IS-IS Protocol 29
 Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface 31
 IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability 34
 Prefix Attribute Flags 34
 IPv4 and IPv6 Source Router ID 35
 Configuring Prefix Attribute N-flag-clear 36
 IS-IS Multi-Domain Prefix SID and Domain Stitching: Example 37
 Configure IS-IS Multi-Domain Prefix SID 38
 Configure Common Router ID 39
 Distribute IS-IS Link-State Data 39

CHAPTER 6

Configure Segment Routing for OSPF Protocol 41
 Enabling Segment Routing for OSPF Protocol 41
 Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface 43
 Configuring an Adjacency SID 45

CHAPTER 7

Configure Segment Routing for BGP 49
 Segment Routing for BGP 49
 Configure BGP Prefix Segment Identifiers 50
 Segment Routing Egress Peer Engineering 51
 Configure Segment Routing Egress Peer Engineering 51
 Configure BGP Link-State 53
 Use Case: Configuring SR-EPE and BGP-LS 55

CHAPTER 8

Configure SR-TE Policies 59
 SR-TE Policy Overview 59
 Usage Guidelines and Limitations 60
 Instantiation of an SR Policy 60
 On-Demand SR Policy – SR On-Demand Next-Hop 60
 SR-ODN Configuration Steps 62

Configuring SR-ODN: Examples	64
Configuring SR-ODN for EVPN-VPWS: Use Case	69
Manually Provisioned SR Policy	90
PCE-Initiated SR Policy	90
SR-TE Policy Path Types	90
Dynamic Paths	91
Optimization Objectives	91
Constraints	92
Configure SR Policy with Dynamic Path	94
Explicit Paths	95
SR-TE Policy with Explicit Path	95
Configuring Explicit Path with Affinity Constraint Validation	98
Protocols	100
Path Computation Element Protocol	100
Configure the Head-End Router as PCEP PCC	100
BGP SR-TE	104
Configure BGP SR Policy Address Family at SR-TE Head-End	104
Traffic Steering	106
Automated Steering	106
Per-Flow Automated Steering	107
Using Binding Segments	111
Stitching SR-TE Polices Using Binding SID: Example	112
L2VPN Preferred Path	116
Policy-Based Tunnel Selection for SR-TE Policy	116
Miscellaneous	117
LDP over Segment Routing Policy	117
Configure Seamless Bidirectional Forwarding Detection	120
Configure the SBFDD Reflector	122
Configure the SBFDD Initiator	123
SR-TE Reoptimization Timers	128
CHAPTER 9	Enabling Segment Routing Flexible Algorithm
	131
	Prerequisites for Flexible Algorithm
	131
	Building Blocks of Segment Routing Flexible Algorithm
	131

- Flexible Algorithm Definition 131
- Flexible Algorithm Membership 132
- Flexible Algorithm Definition Advertisement 132
- Flexible Algorithm Prefix-SID Advertisement 132
- Calculation of Flexible Algorithm Path 132
- Installation of Forwarding Entries for Flexible Algorithm Paths 133
- Configuring Flexible Algorithm 133
- Example: Configuring IS-IS Flexible Algorithm 135
- Example: Traffic Steering to Flexible Algorithm Paths 135
 - BGP Routes on PE – Color Based Steering 135

CHAPTER 10 **Configure Segment Routing Path Computation Element 141**

- About SR-PCE 141
- Usage Guidelines and Limitations 142
- Configure SR-PCE 142
 - Configure the Disjoint Policy (Optional) 144
- PCE-Initiated SR Policies 145

CHAPTER 11 **Configure Performance Measurement 149**

- Liveness Monitoring 150
- Delay Measurement 150
 - Measurement Modes 151
 - Link Delay Measurement 153
 - Delay Measurement for IP Endpoint 164
 - IP Endpoint Delay Measurement over MPLS Network Usecases 165

CHAPTER 12 **Configure Topology-Independent Loop-Free Alternate (TI-LFA) 175**

- Usage Guidelines and Limitations 176
- Configuring TI-LFA for IS-IS 177
- Configuring TI-LFA for OSPF 179

CHAPTER 13 **Configure Segment Routing Microloop Avoidance 181**

- About Segment Routing Microloop Avoidance 181
- Configure Segment Routing Microloop Avoidance for IS-IS 183

Configure Segment Routing Microloop Avoidance for OSPF 184

CHAPTER 14

Configure Segment Routing Mapping Server 187

Segment Routing Mapping Server 187

Usage Guidelines and Restrictions 188

Segment Routing and LDP Interoperability 188

Example: Segment Routing LDP Interoperability 188

Configuring Mapping Server 191

Enable Mapping Advertisement 193

Configure Mapping Advertisement for IS-IS 193

Configure Mapping Advertisement for OSPF 194

Enable Mapping Client 195

CHAPTER 15

Using Segment Routing Traffic Matrix 197

Segment Routing Traffic Matrix 197

Traffic Collector Process 197

Configuring Traffic Collector 198

Displaying Traffic Information 200

CHAPTER 16

Using Segment Routing OAM 203

MPLS Ping and Traceroute for BGP and IGP Prefix-SID 203

Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID 204

MPLS LSP Ping and Traceroute Nil FEC Target 205

Examples: LSP Ping and Traceroute for Nil_FEC Target 206

Segment Routing Ping and Traceroute 207

Segment Routing Ping 207

Segment Routing Traceroute 209

Segment Routing Data Plane Monitoring 212

Configure SR DPM 215



Preface



Note This release has reached end-of-life status. For more information, see the [End-of-Life and End-of-Sale Notices](#).

The *Segment Routing Configuration Guide for Cisco NCS 6000 Series Routers* preface contains these sections:

- [Changes to This Document, on page ix](#)
- [Communications, Services, and Additional Information, on page ix](#)

Changes to This Document

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

Date	Change Summary
February 2022	Republished for Release 7.4.2
July 2021	Initial release of this document

Communications, Services, and Additional Information

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

Cisco Bug Search Tool

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



CHAPTER 1

New and Changed Information for Segment Routing Features

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

- [New and Changed Segment Routing Features, on page 1](#)

New and Changed Segment Routing Features

Segment Routing Features Added or Modified in IOS XR Release 7.4.x

Feature	Description	Introduced/Changed in Release	Where Documented
Segment Routing over IPv6 (SRv6)	This feature is introduced.	Release 7.4.2	Configure Segment Routing over IPv6 (SRv6)



CHAPTER 2

About Segment Routing

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

- [Scope, on page 3](#)
- [Need, on page 4](#)
- [Benefits, on page 4](#)
- [Workflow for Deploying Segment Routing, on page 5](#)

Scope

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

Segments

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

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

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

- An adjacency segment is identified by a label called an adjacency SID, which represents a specific adjacency, such as egress interface, to a neighboring router. 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 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 .

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 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 TI-LFA and Microloop Avoidance
6. Configure the Segment Routing Mapping Server
7. Collect Traffic Statistics



CHAPTER 3

Configure Segment Routing over IPv6 (SRv6)

Table 1: Feature History Table

Feature Name	Release	Description
SRv6 with Micro-Segment (uSID)	Release 7.4.2	<p>This release introduces support for Segment Routing over IPv6 data plane using Micro SIDs (uSIDs).</p> <p>This feature allows the source router to encode multiple SRv6 uSID instructions within a single 128-bit SID address. Such functionality allows for efficient and compact SRv6 SID representation with a low MTU overhead.</p> <p>The feature introduces the following commands:</p> <ul style="list-style-type: none">• segment-routing srv6 locators locator• segment-routing srv6 logging locator status• segment-routing srv6 sid holdtime• show segment-routing srv6 locator• show segment-routing srv6 manager

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

- [Segment Routing over IPv6 Overview, on page 7](#)
- [SRv6 Micro-Segment \(uSID\), on page 12](#)
- [Usage Guidelines and Limitations, on page 20](#)
- [Configuring SRv6, on page 21](#)
- [Configuring SRv6 under IS-IS, on page 23](#)

Segment Routing over IPv6 Overview

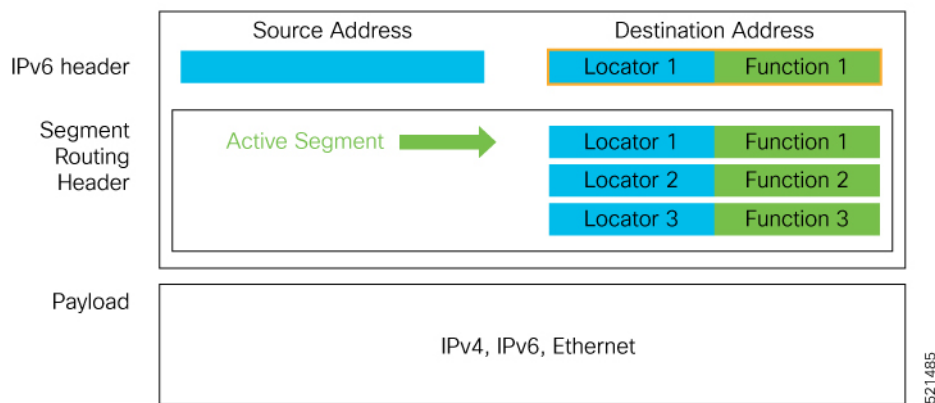
Segment Routing (SR) can be applied on both MPLS and IPv6 data planes. Segment Routing over IPv6 (SRv6) extends Segment Routing support with IPv6 data plane.

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.

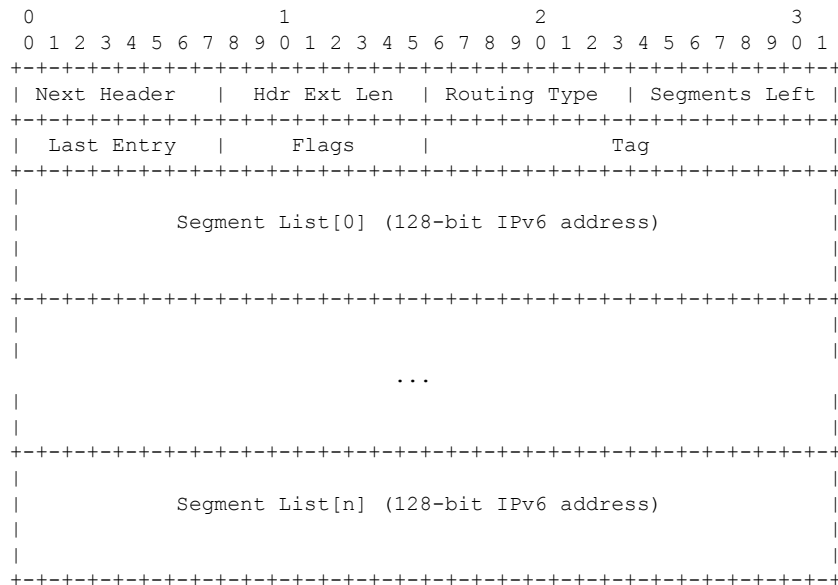
Figure 1: Network Program in the Packet Header



521485

The SRv6 SRH is documented in IETF RFC [IPv6 Segment Routing Header \(SRH\)](#).

The SRH is defined as follows:



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

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 $8 * (\text{Hdr Ext Len} + 1)$
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.

SRv6 Micro-Segment (uSID)

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 "*Fbbuu*", 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.

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
- uA—A short notation for the NEXT-CSID End.X behavior with a pseudocode of shift-and-xconnect
- 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

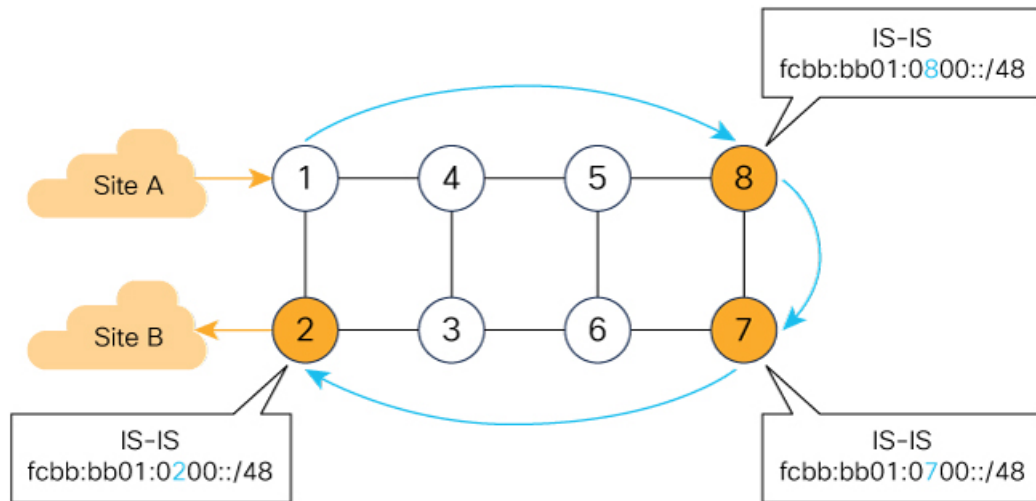
For example:

- 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



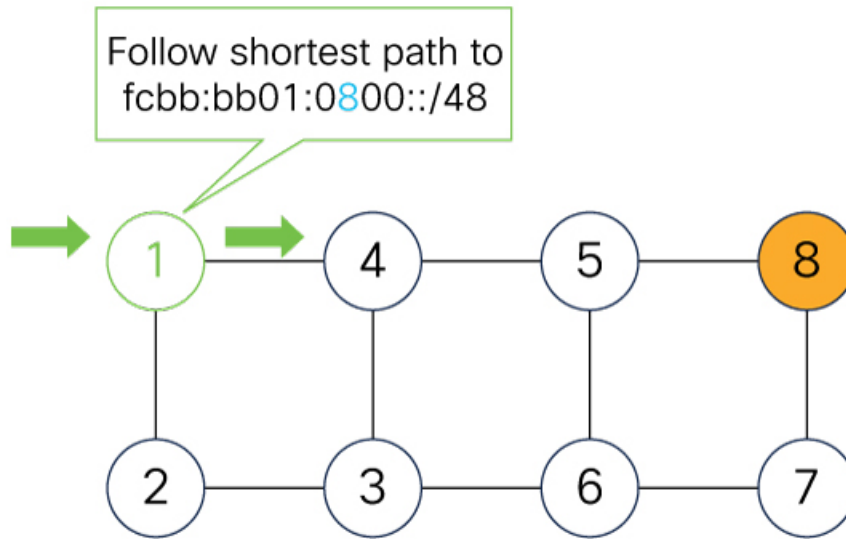
- Node 1 encapsulates IPv4 packet from Site A and sends an IPv6 packet with DA = `fcbb:bb01:0800:0700:0200:f001:0000:0000`
- Traffic engineered path via 8 and 7 using a single 128-bit SRv6 SID
- One single micro-program in the DA is enough

521410

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) advertised by Node 2 for the VPNv4 service

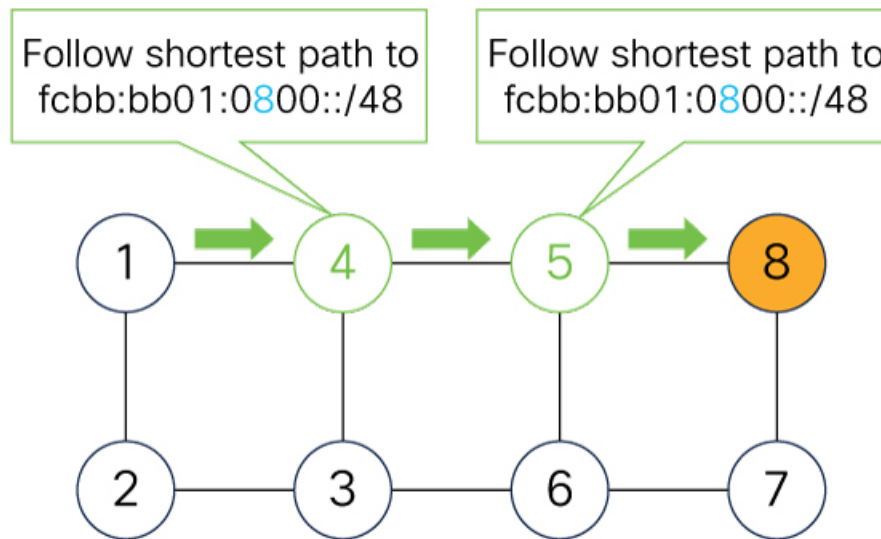
Figure 3: Node 1: End.B6.Encaps Behavior



DA = fcbb:bb01:0800:0700:0200:f001:0000:0000

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



DA = fcbb:bb01:0800:0700:0200:f001:0000:0000

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:

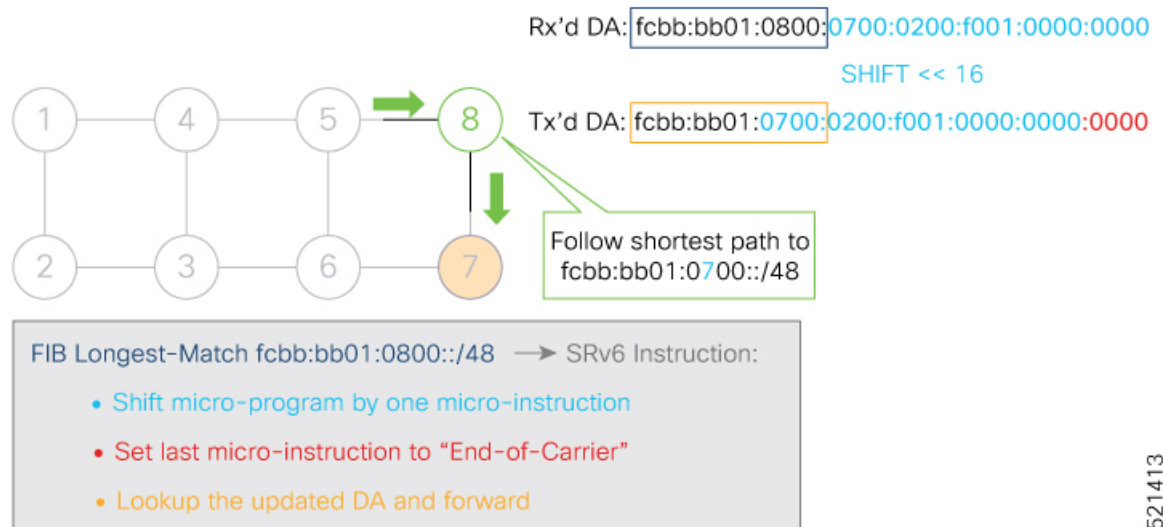
1. Pops its own uSID (0800)

521411

521412

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

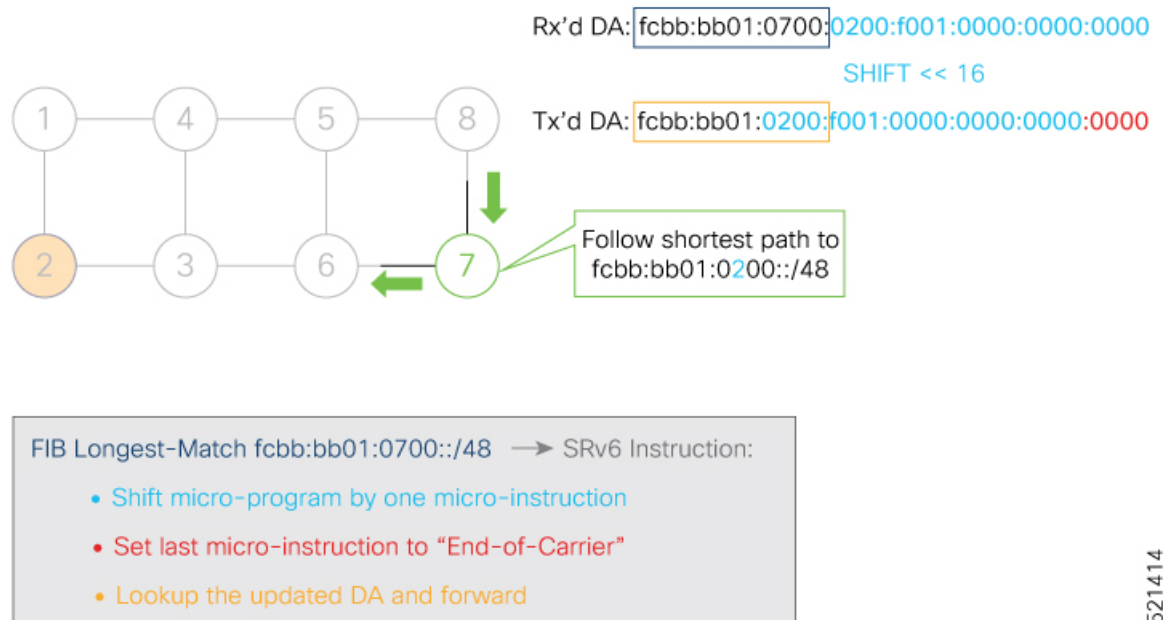
Figure 5: Node 8: SRv6 uN Behavior (Shift and Forward)



521413

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

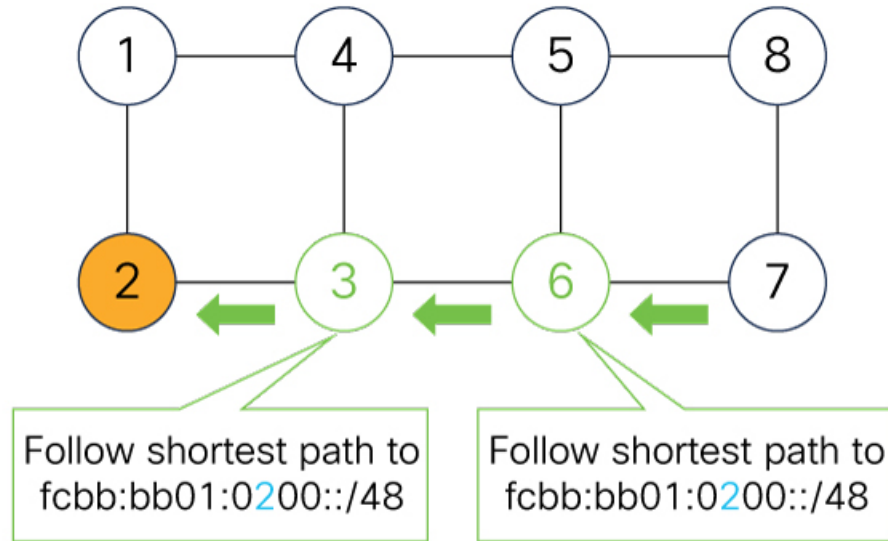
Figure 6: Node 7: SRv6 uN Behavior (Shift and Forward)



521414

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

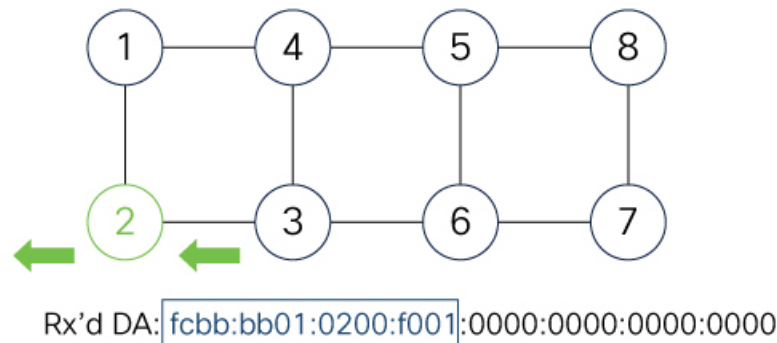


521415

DA = fcbb:bb01:0200:f001:0000:0000:0000:0000

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



FIB Longest-Match fcbb:bb01:0200:f001::/64 → SRv6 Instruction:

- Decapsulate and Lookup of inner IPv4 packet

521416

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:f001:0000:0000:0000:0000
- @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.
- 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 7.4.2 supports the following SRv6 uSID behaviors:
 - uN (shift)
 - uA (shift)
- SRv6 over GRE interface is not supported
- SRv6 over BVI interface is not supported

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
 - FCBB: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
 - FCBB: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 FC00:0100/32 to FC00:013F/32)

Platform-Specific Guidelines and Limitations

This release supports SRv6 on NCS 6000 series routers.

- The NCS 6000 series router can act as transit router on the SRv6 uSID network, performing uN/uA (shift/lookup) operation.
- The following line cards are supported:
 - NC6-60x10GE-L-S, NC6-60x10GE-M-S
 - NC6-10X100G-L-K, NC6-10X100G-L-P, NC6-10X100G-M-P, NC6-10X100G-M-K
 - NC6-20X100GE-L-C, NC6-20X100GE-M-C
- The NCS 6000 series router cannot act as service PE
- PSP, USP behaviors for uN and uA are not supported
- TI-LFA is not supported
- Microloop Avoidance is not supported

Configuring SRv6

Enabling SRv6 involves the following high-level configuration steps:

- Configure SRv6 locator(s)
- Enable SRv6 under IS-IS

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.

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)# prefix 2001:0:8::/48
```

(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.

The following example shows how to configure optional SRv6 parameters:

```
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) #
```

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: 1:1:1::1
    Hop-Limit: Default
    Traffic-class: Default
Summary:
  Number of Locators: 5 (5 operational)
  Number of SIDs: 12 (0 stale)
  Max SIDs: 9000
  OOR:
    Thresholds: Green 450, Warning 270
    Status: Resource Available
      History: (0 cleared, 0 warnings, 0 full)
    Block 2001::/32:
      Number of SIDs free: 7674
      Max SIDs: 7680
      Thresholds: Green 384, Warning 231
      Status: Resource Available
      History: (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: No
  Microloop-Avoidance: No
  Endpoint behaviors:
    uN (shift)
    uA (shift)
  Headend behaviors:
    None
  Security rules:
    None
  Counters:
    None
  Signaled parameters:
    Max-SL : 0
    Max-End-Pop-SRH : 0
    Max-H-Insert : 0 sids
    Max-H-Encap : 0 sids
```



```

Max-End-D      : 0
Configurable parameters (under srv6):
Encapsulation:
  Source Address: No
  Hop-Limit     : value=No, propagate=No
  Traffic-class : value=No, propagate=No
Max SIDs: 9000
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        2        0     2001:0:8::/48   Up      U
(U): Micro-segment (behavior: uN (shift))
Interface:
  Name: srv6-myLoc1
  IFH : 0x0800002c
  IPv6 address: 2001:0:8::/48
  Number of SIDs: 1
  Created: Jan 11 14:22:30.141 (2w5d 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 (shift)  '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 (shift)  'default':1     sidmgr         InUse  Y
  SID Function: 0x1
  SID context: { table-id=0xe0800000 ('default':IPv6/Unicast), opaque-id=1 }
  Locator: 'myLoc1'
  Allocation type: Dynamic
  Created: Jan 11 14:22:30.490 (2w5d 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 following command:

- **router isis** *instance* **address-family ipv6 unicast segment-routing srv6 locator** *locator* [**level** {**1** | **2**}]—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.

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 6000 Series Routers*.



CHAPTER 4

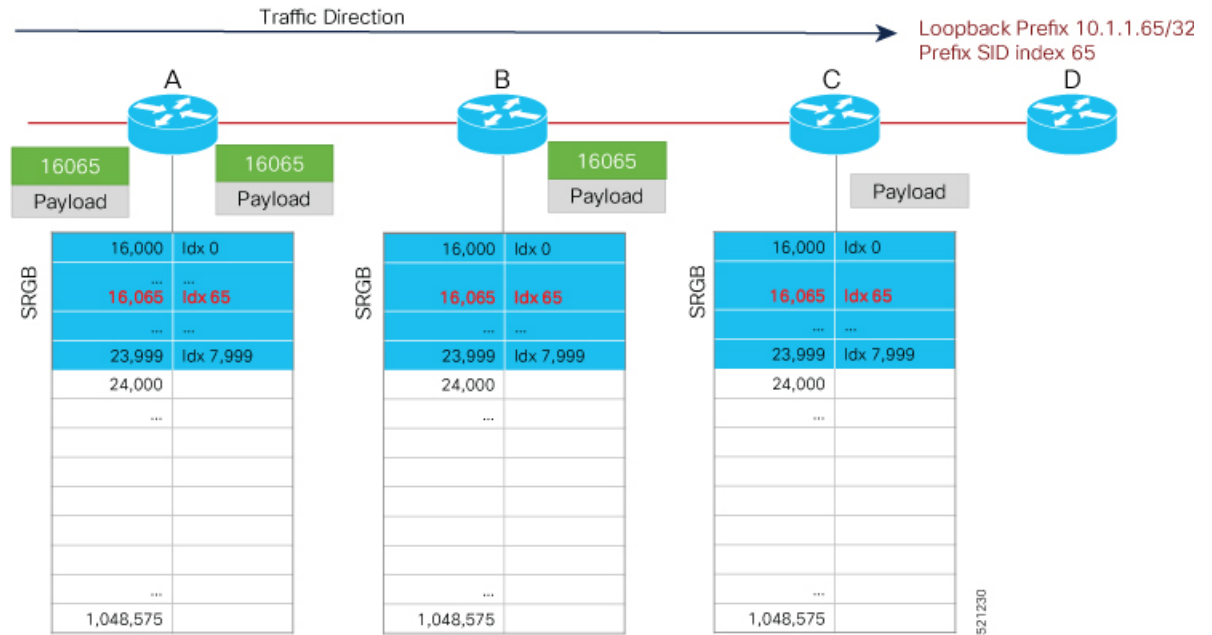
Configure Segment Routing Global Block

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

- [About the Segment Routing Global Block, on page 25](#)
- [Setup a Non-Default Segment Routing Global Block Range, on page 27](#)

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.

Setup a Non-Default Segment Routing Global Block Range

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

SUMMARY STEPS

1. **configure**
2. **segment-routing global-block** *starting_value ending_value*
3. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	segment-routing global-block <i>starting_value ending_value</i> Example: RP/0/RP0/CPU0:router(config)# <code>segment-routing global-block 16000 80000</code>	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. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • No —Exits the configuration session without committing the configuration changes. • 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.



CHAPTER 5

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 Cisco NCS 6000 Series Routers, see the *Implementing IS-IS* module in the *Routing Configuration Guide for Cisco NCS 6000 Series Routers*.

- [Enabling Segment Routing for IS-IS Protocol, on page 29](#)
- [Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 31](#)
- [IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability, on page 34](#)
- [IS-IS Multi-Domain Prefix SID and Domain Stitching: Example, on page 37](#)

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.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } [**unicast**]
4. **metric-style wide** [**level** { **1** | **2** }]
5. **router-id loopback** *loopback interface used for prefix-sid*
6. **segment-routing mpls** [**sr-prefer**]
7. **exit**
8. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	router isis <i>instance-id</i> Example: RP/0/RP0/CPU0:router(config)# router isis <i>isp</i>	Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode. Note You can change the level of routing to be performed by a particular routing instance by using the is-type router configuration command.
Step 3	address-family { ipv4 ipv6 } [unicast] Example: RP/0/RP0/CPU0:router(config-isis)# address-family <i>ipv4 unicast</i>	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.
Step 4	metric-style wide [level { 1 2 }] Example: RP/0/RP0/CPU0:router(config-isis-af)# metric-style <i>wide level 1</i>	Configures a router to generate and accept only wide link metrics in the Level 1 area.
Step 5	router-id loopback <i>loopback interface used for prefix-sid</i> Example: RP/0/RP0/CPU0:router(config-isis-af)# router-id <i>loopback0</i>	Configures router ID for each address-family (IPv4/IPv6). 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]	Segment routing is enabled by the following actions:

	Command or Action	Purpose
	<p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis-af)# segment-routing mpls</pre>	<ul style="list-style-type: none"> • MPLS forwarding is enabled on all interfaces where IS-IS is active. • All known prefix-SIDs in the forwarding plain are programmed, with the prefix-SIDs advertised by remote routers or learned through local or remote mapping server. • The prefix-SIDs locally configured are advertised. <p>Use the sr-prefer keyword to set the preference of segment routing (SR) labels over label distribution protocol (LDP) labels.</p>
Step 7	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis-af)# exit RP/0/RP0/CPU0:router(config-isis)# exit</pre>	
Step 8	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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.

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



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

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

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

Before you begin

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	router isis <i>instance-id</i> Example: 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. <ul style="list-style-type: none"> • 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 <i>instance</i> Example: RP/0/RP0/CPU0:router(config-isis)# interface Loopback0	Specifies the loopback interface and instance.

	Command or Action	Purpose
Step 4	<p>address-family { ipv4 ipv6 } [unicast]</p> <p>Example:</p> <p>The following is an example for ipv4 address family:</p> <pre>RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast</pre>	<p>Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.</p>
Step 5	<p>prefix-sid [strict-spf] {index <i>SID-index</i> absolute <i>SID-value</i>} [n-flag-clear] [explicit-null]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# prefix-sid index 1001</pre> <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# prefix-sid strict-spf index 101</pre> <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# prefix-sid absolute 17001</pre>	<p>Configures the prefix-SID index or absolute value for the interface.</p> <p>Specify strict-spf to configure the prefix-SID to use the SPF path instead of the SR-TE .</p> <p>Specify index <i>SID-index</i> for each node to create a prefix SID based on the lower boundary of the SRGB + the index.</p> <p>Specify absolute <i>SID-value</i> for each node to create a specific prefix SID within the SRGB.</p> <p>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).</p> <p>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.</p> <p>Note IS-IS does not advertise separate explicit-NUL or flags for regular SIDs and strict-SPF SIDs. The settings in the regular SID are used if the settings are different.</p>
Step 6	<p>Use the commit or end command.</p>	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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
  Hostname:      router
  IP Address:    10.0.0.1
  IPv6 Address:  2001:0db8:1234::0a00:0001
  Router Cap:    10.0.0.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
    SR Algorithm:
      Algorithm: 0
      Algorithm: 1
<...>
Metric: 0           IP-Extended 10.0.0.1/32
  Prefix-SID Index: 1001, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
  Prefix-SID Index: 101, Algorithm:1, R:0 N:1 P:0 E:0 V:0 L:0
<...>

```

IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability

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

- Prefix Attribute Flags
- IPv4 and IPv6 Source Router ID

Prefix Attribute Flags

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

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



Note Prefix attributes are only added when wide metric is used.

Prefix Attribute Flags Sub-TLV Format

```

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

```

Prefix Attribute Flags Sub-TLV Fields

Field	Description
X (External Prefix Flag)	This flag is set if the prefix has been redistributed from another protocol. The value of the flag is preserved when the prefix is propagated to another level.
R (Re-advertisement Flag)	This flag is set to 1 by the Level 1-2 router when the prefix is propagated between IS-IS levels (from Level 1 to Level 2, or from Level 2 to Level 1). This flag is set to 0 when the prefix is connected locally to an IS-IS-enabled interface (regardless of the level configured on the interface).
N (Node Flag)	For prefixes that are propagated from another level: <ol style="list-style-type: none"> 1. Copy the N-flag from the prefix attribute sub-TLV, if present in the source level. 2. Copy the N-flag from the prefix-SID sub-TLV, if present in the source level. 3. Otherwise, set to 0. For connected prefixes: <ol style="list-style-type: none"> 1. Set to 0 if prefix-attributes n-flag-clear is configured (see Configuring Prefix Attribute N-flag-clear). 2. Set to 0 if n-flag-clear { n-flag-clearSID-index n-flag-clearSID-value } n-flag-clear is configured (see Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface). 3. Otherwise, set to 1 when the prefix is a host prefix (/32 for IPv4, /128 for IPv6) that is associated with a loopback address. <p>Note If the flag is set and the prefix length is not a host prefix, then the flag must be ignored.</p>

IPv4 and IPv6 Source Router ID

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

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

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

The source router ID is propagated between levels.

Table 2: Source Router Sub-TLV Format

IPv4 Source Router ID	Type: 11 Length: 4 Value: IPv4 Router ID of the source of the prefix advertisement
IPv6 Source Router ID	Type: 12 Length: 16 Value: IPv6 Router ID of the source of the prefix advertisement

Configuring Prefix Attribute N-flag-clear

The N-flag is set to 1 when the prefix is a host prefix (/32 for IPV4, /128 for IPV6) that is associated with a loopback address. The advertising router can be configured to not set this flag. This task explains how to clear the N-flag.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **interface Loopback** *instance*
4. **prefix-attributes n-flag-clear** [Level-1 | Level-2]
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	router isis <i>instance-id</i> Example: RP/0/RP0/CPU0:router(config)# router isis 1	
Step 3	interface Loopback <i>instance</i> Example: RP/0/RP0/CPU0:router(config)# interface Loopback0	Specifies the loopback interface.
Step 4	prefix-attributes n-flag-clear [Level-1 Level-2] Example:	Clears the prefix attribute N-flag explicitly.

	Command or Action	Purpose
	RP/0/RP0/CPU0:router(config-if)# isis prefix-attributes n-flag-clear	
Step 5	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Verify the prefix attribute configuration:

```
RP/0/RP0/CPU0:router# show isis database verbose

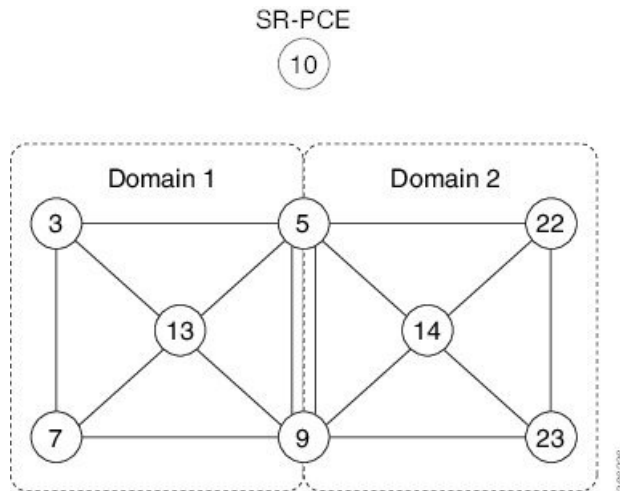
IS-IS 1 (Level-2) Link State Database
LSPID          LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
router.00-00   * 0x0000039b  0xfc27        1079          0/0/0
  Area Address: 49.0001
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6 Unicast                      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
    Algorithm: 1
<...>
  Metric: 0      IP-Extended 10.0.0.1/32
    Prefix-SID Index: 1001, Algorithm:0, R:1 N:0 P:1 E:0 V:0 L:0
    Prefix Attribute Flags: X:0 R:1 N:0
  Metric: 10     IP-Extended 10.0.0.2/32
    Prefix-SID Index: 1002, Algorithm:0, R:0 N:1 P:0 E:0 V:0 L:0
    Prefix Attribute Flags: X:0 R:0 N:1
  Source Router ID: 10.0.0.2
<...>
```

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

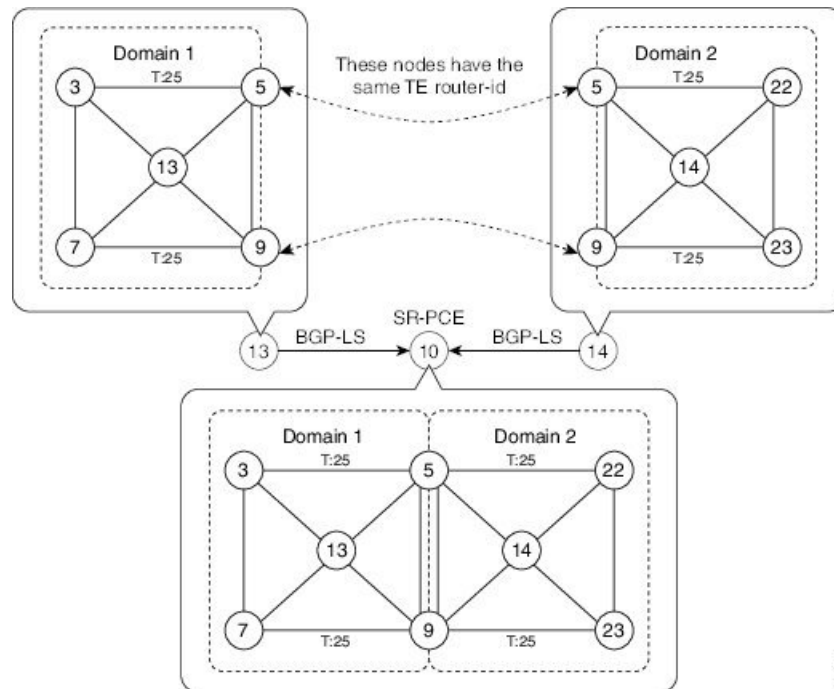
router isis Domain2
 address-family ipv4 unicast
  router-id loopback0
```

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.



CHAPTER 6

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 Cisco NCS 6000 Series Routers, see the *Implementing OSPF* module in the *Routing Configuration Guide for Cisco NCS 6000 Series Routers*.

- [Enabling Segment Routing for OSPF Protocol, on page 41](#)
- [Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 43](#)
- [Configuring an Adjacency SID, on page 45](#)

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.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **segment-routing mpls**
4. **segment-routing sr-prefer**
5. **area** *area*
6. **segment-routing mpls**
7. **exit**
8. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/RP0/CPU0:router (config)# <code>router ospf 1</code>	Enables OSPF routing for the specified routing process and places the router in router configuration mode.
Step 3	segment-routing mpls Example: RP/0/RP0/CPU0:router (config-ospf)# <code>segment-routing mpls</code>	Enables segment routing using the MPLS data plane on the routing process and all areas and interfaces in the routing process. Enables segment routing forwarding on all interfaces in the routing process and installs the SIDs received by OSPF in the forwarding table.
Step 4	segment-routing sr-prefer Example: RP/0/RP0/CPU0:router (config-ospf)# <code>segment-routing sr-prefer</code>	Sets the preference of segment routing (SR) labels over label distribution protocol (LDP) labels.
Step 5	area <i>area</i> Example: RP/0/RP0/CPU0:router (config-ospf)# <code>area 0</code>	Enters area configuration mode.
Step 6	segment-routing mpls Example:	(Optional) Enables segment routing using the MPLS data plane on the area and all interfaces in the area. Enables

	Command or Action	Purpose
	<code>RP/0/RP0/CPU0:router(config-ospf-ar)# segment-routing mpls</code>	segment routing forwarding on all interfaces in the area and installs the SIDs received by OSPF in the forwarding table.
Step 7	<p>exit</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ospf-ar)# exit RP/0/RP0/CPU0:router(config-ospf)# exit</pre>	
Step 8	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *value*
4. **interface Loopback** *interface-instance*
5. **prefix-sid** [**strict-spf**] {**index** *SID-index* | **absolute** *SID-value* } [**n-flag-clear**] [**explicit-null**]
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	router ospf <i>process-name</i> Example: 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 <i>value</i> Example: RP/0/RP0/CPU0:router (config-ospf)# area 0	Enters area configuration mode.
Step 4	interface Loopback <i>interface-instance</i> Example: RP/0/RP0/CPU0:router (config-ospf-ar)# interface Loopback0 passive	Specifies the loopback interface and instance.
Step 5	prefix-sid [strict-spf] { index <i>SID-index</i> absolute <i>SID-value</i> } [n-flag-clear] [explicit-null] Example: RP/0/RP0/CPU0:router (config-ospf-ar)# prefix-sid index 1001 RP/0/RP0/CPU0:router (config-ospf-ar)# prefix-sid absolute 17001	<p>Configures the prefix-SID index or absolute value for the interface.</p> <p>Specify strict-spf to configure the prefix-SID to use the SPF path instead of the SR-TE tunnel.</p> <p>Specify index <i>SID-index</i> for each node to create a prefix SID based on the lower boundary of the SRGB + the index.</p> <p>Specify absolute <i>SID-value</i> for each node to create a specific prefix SID within the SRGB.</p> <p>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).</p>

	Command or Action	Purpose
		To disable penultimate-hop-popping (PHP) and add an explicit-Null label, enter the <code>explicit-null</code> keyword. OSPF sets the <code>E</code> flag in the prefix-SID sub TLV.
Step 6	Use the <code>commit</code> or <code>end</code> command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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
```

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 OSPF Adj-SID sub-TLV. The P-flag is defined for manually allocated Adj-SIDs.

```

 0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|B|V|I|G|P|   |
+---+---+---+---+---+---+

```

Table 3: Adjacency Segment Identifier (Adj-SID) Flags Sub-TLV Fields

Field	Description
P (Persistent)	This flag is set if the Adj-SID is persistent (manually allocated).

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

Before you begin

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

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

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area*
4. **interface** *type interface-path-id*
5. **adjacency-sid** {*index adj-SID-index* | **absolute** *adj-SID-value*} [**protected**]
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	router ospf <i>process-name</i> Example:	Enables OSPF routing for the specified routing instance, and places the router in router configuration mode.

	Command or Action	Purpose
	<pre>RP/0/RP0/CPU0:router(config)# router ospf 1</pre>	
Step 3	<p>area <i>area</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ospf)# area 0</pre>	Enters area configuration mode.
Step 4	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ospf-ar)# interface HundredGigE0/0/0/1</pre>	Specifies the interface and enters interface configuration mode.
Step 5	<p>adjacency-sid {index <i>adj-SID-index</i> absolute <i>adj-SID-value</i>} [protected]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-config-ospf-ar-if)# adjacency-sid index 10</pre> <pre>RP/0/RP0/CPU0:router(config-config-ospf-ar-if)# adjacency-sid absolute 15010</pre>	<p>Configures the Adj-SID index or absolute value for the interface.</p> <p>Specify index <i>adj-SID-index</i> for each link to create an Adj-SID based on the lower boundary of the SRLB + the index.</p> <p>Specify absolute <i>adj-SID-value</i> for each link to create a specific Adj-SID within the SRLB.</p> <p>Specify if the Adj-SID is protected. For each primary path, if the Adj-SID is protected on the primary interface and a backup path is available, a backup path is installed. By default, manual Adj-SIDs are not protected.</p>
Step 6	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

What to do next

Configure the SR-TE policy.



CHAPTER 7

Configure Segment Routing for BGP

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

This module provides the configuration information used to enable Segment Routing for BGP.



Note For additional information on implementing BGP on your Cisco NCS 6000 Series Routers, see the *Implementing BGP* module in the *Routing Configuration Guide for Cisco NCS 6000 Series Routers*.

- [Segment Routing for BGP, on page 49](#)
- [Configure BGP Prefix Segment Identifiers, on page 50](#)
- [Segment Routing Egress Peer Engineering, on page 51](#)
- [Configure BGP Link-State, on page 53](#)
- [Use Case: Configuring SR-EPE and BGP-LS, on page 55](#)

Segment Routing for BGP

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

In an MPLS-based DC fabric, the eBGP sessions between the nodes exchange BGP labeled unicast (BGP-LU) network layer reachability information (NLRI). An MPLS-based DC fabric allows any leaf (top-of-rack or border router) in the fabric to communicate with any other leaf using a single label, which results in higher packet forwarding performance and lower encapsulation overhead than traditional BGP-based DC fabric. However, since each label value might be different for each hop, an MPLS-based DC fabric is more difficult to troubleshoot and more complex to configure.

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

Configure BGP Prefix Segment Identifiers

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

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



Note 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 6000 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/RSP0/CPU0:router(config)# segment-routing global-block 16000 23999

RP/0/RSP0/CPU0:router(config)# route-policy SID($SID)
RP/0/RSP0/CPU0:router(config-rpl)# set label-index $SID
RP/0/RSP0/CPU0:router(config-rpl)# end policy

RP/0/RSP0/CPU0:router(config)# router bgp 1
RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 10.1.1.1
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)# network 10.1.1.3/32 route-policy SID(3)
RP/0/RSP0/CPU0:router(config-bgp-af)# allocate-label all
```

```

RP/0/RSP0/CPU0:router(config-bgp-af)# commit
RP/0/RSP0/CPU0:router(config-bgp-af)# end

RP/0/RSP0/CPU0:router# show bgp 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.

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.

SUMMARY STEPS

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

5. **exit**
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>router bgp <i>as-number</i></p> <p>Example:</p> <pre>RP/0/RSP0/CPU0:router(config)# router bgp 1</pre>	Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Step 2	<p>neighbor <i>ip-address</i></p> <p>Example:</p> <pre>RP/0/RSP0/CPU0:router(config-bgp)# neighbor 192.168.1.3</pre>	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.
Step 3	<p>remote-as <i>as-number</i></p> <p>Example:</p> <pre>RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 3</pre>	Creates a neighbor and assigns a remote autonomous system number to it.
Step 4	<p>egress-engineering</p> <p>Example:</p> <pre>RP/0/RSP0/CPU0:router(config-bgp-nbr)# egress-engineering</pre>	Configures the egress node with EPE for the eBGP peer.
Step 5	<p>exit</p> <p>Example:</p> <pre>RP/0/RSP0/CPU0:router(config-bgp-nbr)# exit RP/0/RSP0/CPU0:router(config-bgp)# exit RP/0/RSP0/CPU0:router(config)#</pre>	
Step 6	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • 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
  !
  !
  !
```

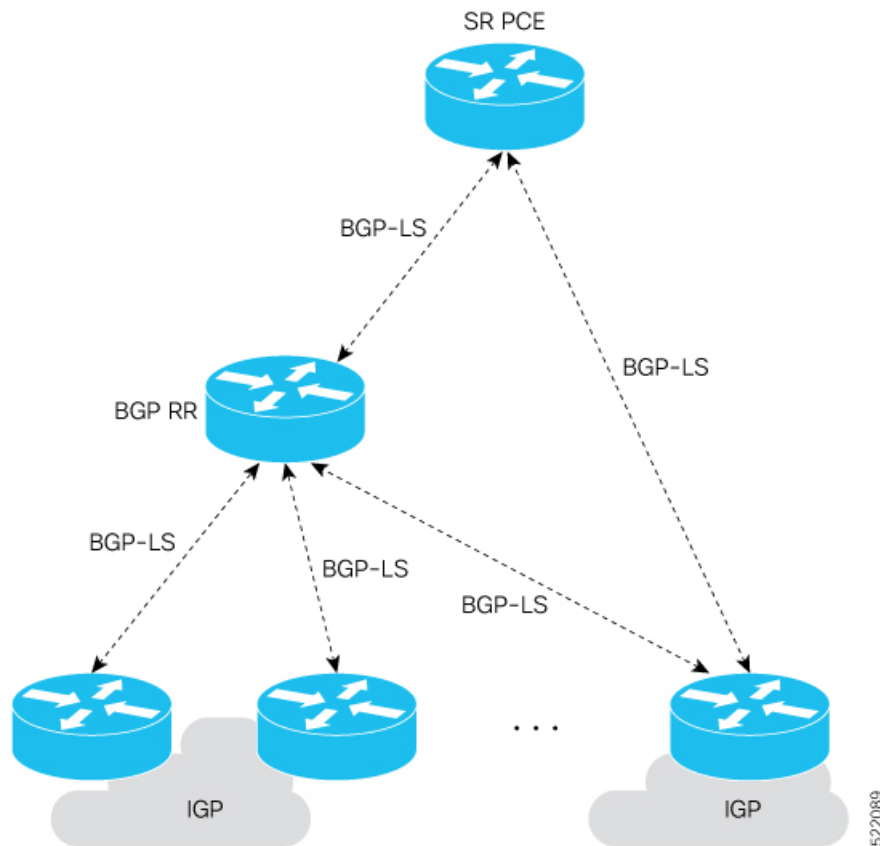
Configure BGP Link-State

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.

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
```

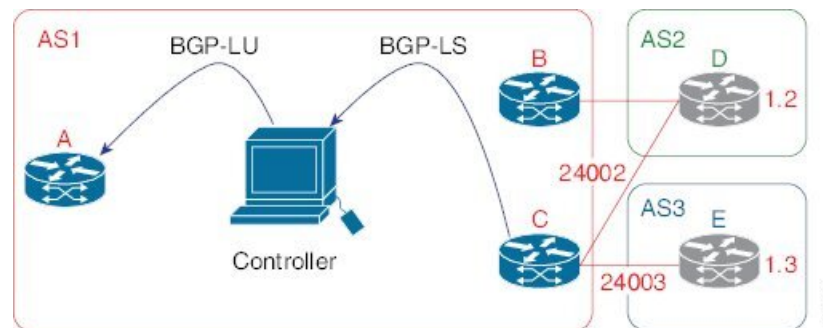
To distribute OSPFv2 link-state data using BGP-LS, use the **distribute link-state** command in router configuration mode.

```
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



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

Example:

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

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

Example:

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

Step 3 Commit the configuration.

Example:

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

Step 4 Verify the configuration.

Example:

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

Egress Engineering Peer Set: 192.168.1.2/32 (10b87210)
  Nexthop: 192.168.1.2
  Version: 2, rn_version: 2
  Flags: 0x00000002
  Local ASN: 1
  Remote ASN: 2
  Local RID: 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/RSP0/CPU0:router_C# show mpls forwarding labels 24002 24003
Local  Outgoing  Prefix          Outgoing      Next Hop      Bytes
Label  Label      or ID          Interface     Interface     Switched
-----
24002  Pop        No ID          Te0/3/0/0    192.168.1.2  0
24003  Pop        No ID          Te0/1/0/0    192.168.1.3  0

```

The output shows that node C installed peer node SIDs in the Forwarding Information Base (FIB).



CHAPTER 8

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, on page 59](#)
- [Usage Guidelines and Limitations, on page 60](#)
- [Instantiation of an SR Policy, on page 60](#)
- [SR-TE Policy Path Types, on page 90](#)
- [Protocols, on page 100](#)
- [Traffic Steering, on page 106](#)
- [Miscellaneous, on page 117](#)

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 [WECCMP]). A candidate path is either dynamic or explicit. See *SR-TE Policy Path Types* section for more information.

Usage Guidelines and Limitations

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.
- 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.

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:

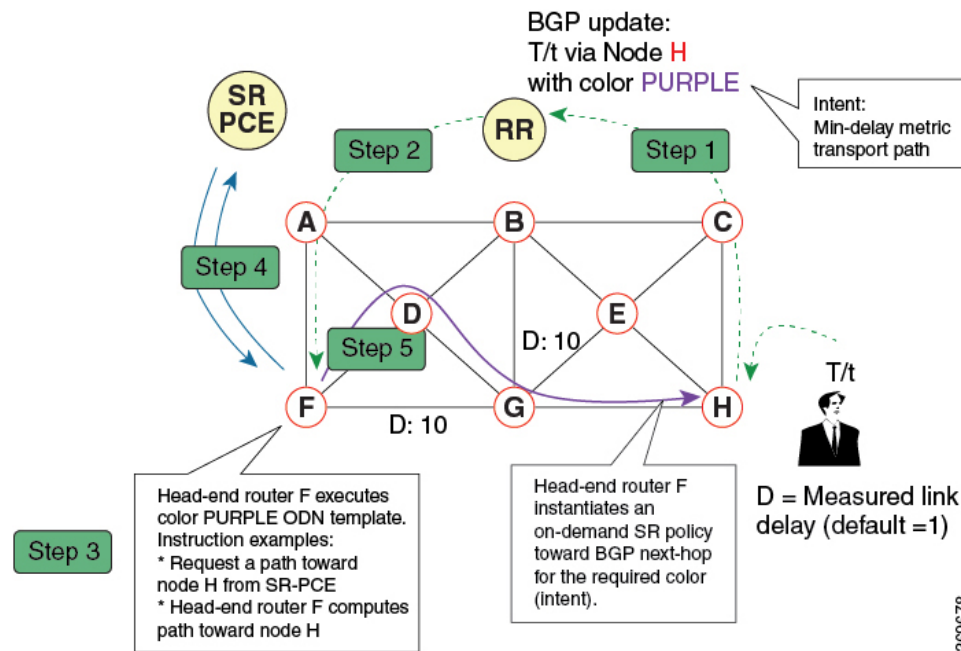
- [On-Demand SR Policy – SR On-Demand Next-Hop](#), on page 60
- [Manually Provisioned SR Policy](#), on page 90
- [PCE-Initiated SR Policy](#), on page 90

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

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

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

The following example shows how SR-ODN works:



1. An egress PE (node H) advertises a BGP route for prefix T/t. This advertisement includes an SLA intent encoded with a BGP color extended community. In this example, the operator assigns color purple (example value = 100) to prefixes that should traverse the network over the delay-optimized path.
2. The route reflector receives the advertised route and advertises it to other PE nodes.
3. Ingress PEs in the network (such as node F) are pre-configured with an ODN template for color purple that provides the node with the steps to follow in case a route with the intended color appears, for example:
 - Contact SR-PCE and request computation for a path toward node H that does not share any nodes with another LSP in the same disjointness group.
 - At the head-end router, compute a path towards node H that minimizes cumulative delay.
4. In this example, the head-end router contacts the SR-PCE and requests computation for a path toward node H that minimizes cumulative delay.
5. After SR-PCE provides the compute path, an intent-driven SR policy is instantiated at the head-end router. Other prefixes with the same intent (color) and destined to the same egress PE can share the same on-demand SR policy. When the last prefix associated with a given [intent, egress PE] pair is withdrawn, the on-demand SR policy is deleted, and resources are freed from the head-end router.

An on-demand SR policy is created dynamically for BGP global or VPN (service) routes. The following services are supported with SR-ODN:

- IPv4 BGP global routes
- IPv6 BGP global routes (6PE)
- VPNv4
- VPNv6 (6vPE)
- EVPN-VPWS (single-homing)

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 142](#).
 - 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 [Routing Configuration Guide for Cisco NCS 6000 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 6000 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 6000 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 ().

```
Router(config-sr-te-color)# maximum-sid-depth 5
```

See [Customize MSD Value at PCC, on page 102](#) 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
```

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
    !
  !
  !
  !
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 State	Oper State	Binding 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           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

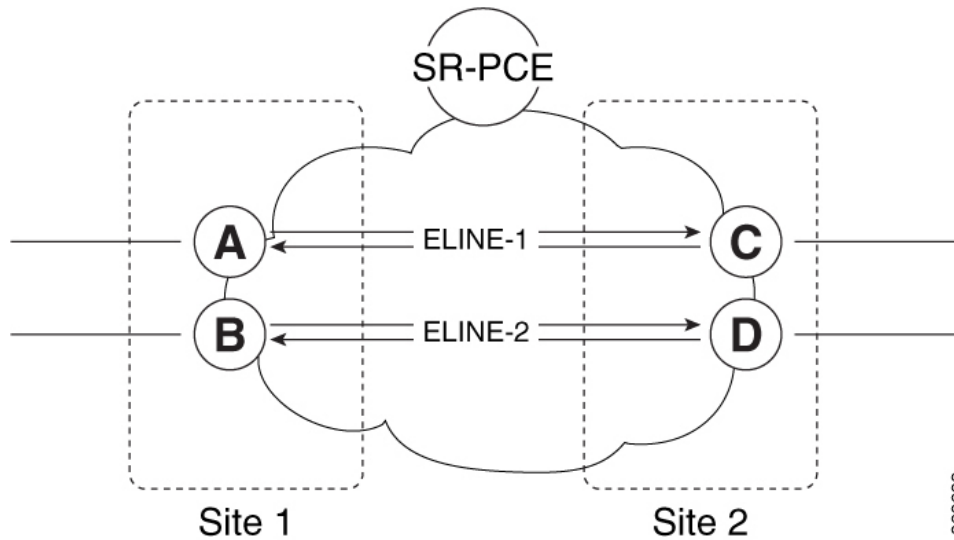
```

Configuring SR-ODN 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



369698

Table 4: Use Case Parameters

IP Addresses of Loopback0 (Lo0) Interfaces	SR-PCE Lo0: 10.1.1.207	
	Site 1: <ul style="list-style-type: none"> • Node A Lo0: 10.1.1.5 • Node B Lo0: 10.1.1.6 	Site 2: <ul style="list-style-type: none"> • Node C Lo0: 10.1.1.2 • Node D Lo0: 10.1.1.4
EVPN-VPWS Service Parameters	ELINE-1: <ul style="list-style-type: none"> • EVPN-VPWS EVI 100 • Node A: AC-ID = 11 • Node C: AC-ID = 21 	ELINE-2: <ul style="list-style-type: none"> • EVPN-VPWS EVI 101 • Node B: AC-ID = 12 • Node D: AC-ID = 22
ODN BGP Color Extended Communities	Site 1 routers (Nodes A and B): <ul style="list-style-type: none"> • set color 10000 • match color 11000 	Site 2 routers (Nodes C and D): <ul style="list-style-type: none"> • set color 11000 • match color 10000
Note	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): <ul style="list-style-type: none"> • group-id = 775 	Site 2 to Site 1 LSPs (from Node C to Node A/from Node D to Node B): <ul style="list-style-type: none"> • group-id = 776

The use case provides configuration and verification outputs for all devices.

Configuration	Verification
Configuration: SR-PCE, on page 71	Verification: SR-PCE, on page 75
Configuration: Site 1 Node A, on page 71	Verification: Site 1 Node A, on page 79
Configuration: Site 1 Node B, on page 72	Verification: Site 1 Node B, on page 82
Configuration: Site 2 Node C, on page 73	Verification: Site 2 Node C, on page 85
Configuration: Site 2 Node D, on page 74	Verification: Site 2 Node D, on page 87

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 144](#) 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 Group	Name	ST	Segment 1 Description	ST	Segment 2 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 State	Oper State	Binding 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 State	Oper State	Binding 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 Group	Name	ST	Segment 1 Description	ST	Segment 2 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 State	Oper State	Binding 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
Group      Name      ST      Segment 1      ST      Segment 2      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 State	Oper State	Binding 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 90](#) 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 145](#) section.

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.

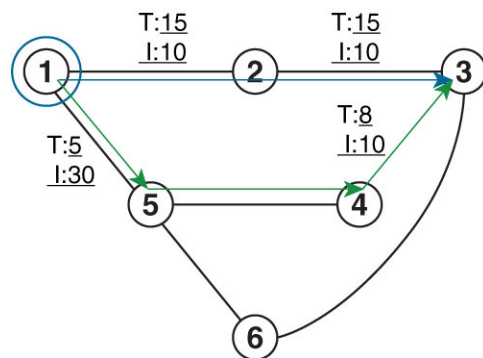
Dynamic Paths

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 92](#) 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:



Default IGP link metric: I:10
 Default TE link metric T:10

520018

- 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 92](#) 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.

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 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

```

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 91](#) section.
2. Define the constraints. See the [Constraints, on page 92](#) section.
3. Create the policy.

Behaviors and Limitations

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
  !
  !
  !
  !
  !

```

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 selects either the protected or the unprotected Adj-SID of the link, depending on the order in which the Adj-SIDs were received.

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 address ipv4 10.1.1.2
Router(config-sr-te-sl)# index 20 address ipv4 10.1.1.3
Router(config-sr-te-sl)# index 30 address ipv4 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 address ipv4 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 address ipv4 10.1.1.2
index 20 address ipv4 10.1.1.3
index 30 address ipv4 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 address ipv4 10.1.1.2
index 20 mpls label 16003
index 30 mpls label 16004
!
segment-list SIDLIST4
index 10 mpls label 16009
index 20 mpls label 16003
index 30 mpls label 16004
!
policy POLICY1
color 10 end-point ipv4 10.1.1.4
candidate-paths
preference 100
explicit segment-list SIDLIST1
!
!
```

```

!
!
policy POLICY2
  color 20 end-point ipv4 10.1.1.4
  candidate-paths
    preference 100
    explicit segment-list SIDLIST1
    !
    preference 200
    explicit segment-list SIDLIST2
    !
  !
!
policy POLICY3
  color 30 end-point ipv4 10.1.1.4
  candidate-paths
    preference 100
    explicit segment-list SIDLIST3
    !
  !
!
!
!
!
!

```

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 address ipv4 10.1.1.2
Router(config-sr-te-sl)# index 20 address ipv4 2.2.2.23
Router(config-sr-te-sl)# index 30 address ipv4 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 address ipv4 10.1.1.2
Router(config-sr-te-sl)# index 30 address ipv4 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 address ipv4 10.1.1.5
Router(config-sr-te-sl)# index 30 address ipv4 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 address ipv4 10.1.1.2
  index 20 address ipv4 2.2.2.23
  index 30 address ipv4 10.1.1.4
  !
 segment-list name SIDLIST2
  index 10 address ipv4 10.1.1.2
  index 30 address ipv4 10.1.1.4
  !
 segment-list name SIDLIST3
  index 10 address ipv4 10.1.1.5
  index 30 address ipv4 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
        !
        !
        !

```

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-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 ().

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

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
```

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:

SUMMARY STEPS

1. **configure**
2. **router bgp** *as-number*
3. **bgp router-id** *ip-address*
4. **address-family** {**ipv4** | **ipv6**} **sr-policy**
5. **exit**
6. **neighbor** *ip-address*
7. **remote-as** *as-number*
8. **address-family** {**ipv4** | **ipv6**} **sr-policy**
9. **route-policy** *route-policy-name* {**in** | **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure	
Step 2	router bgp <i>as-number</i> Example: RP/0/RSP0/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 <i>ip-address</i> Example: RP/0/RSP0/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 Example: RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 sr-policy	Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.
Step 5	exit	
Step 6	neighbor <i>ip-address</i> Example: RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.0.1	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.
Step 7	remote-as <i>as-number</i> Example: RP/0/RSP0/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 Example: RP/0/RSP0/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 <i>route-policy-name</i> { in out } Example: RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#	Applies the specified policy to IPv4 or IPv6 unicast routes.

	Command or Action	Purpose
	route-policy pass out	

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 IPv6 SR policies.

```
router bgp 65000
  bgp router-id 10.1.1.1
  address-family ipv6 sr-policy
  !
  neighbor 3001::10:1:3:1
    remote-as 10
    description *** eBGP session to BGP SRTE controller ***
    address-family ipv6 sr-policy
      route-policy pass in
      route-policy pass out
    !
  !
  !
```

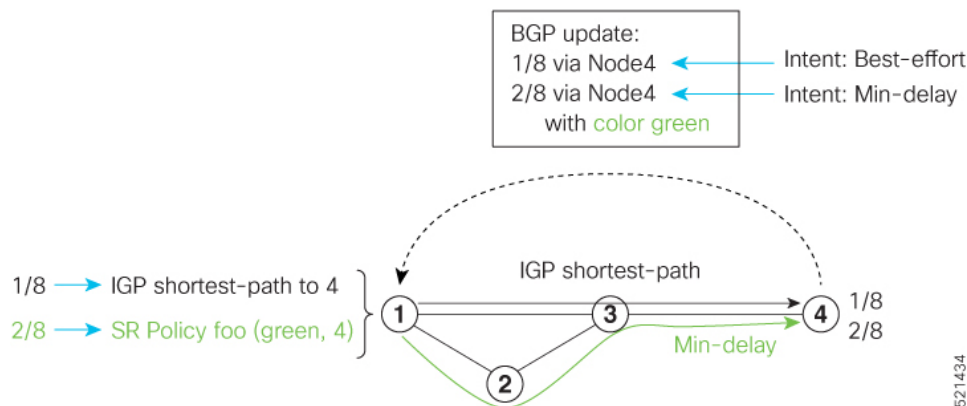
Traffic Steering

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

Per-Flow Automated Steering

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).

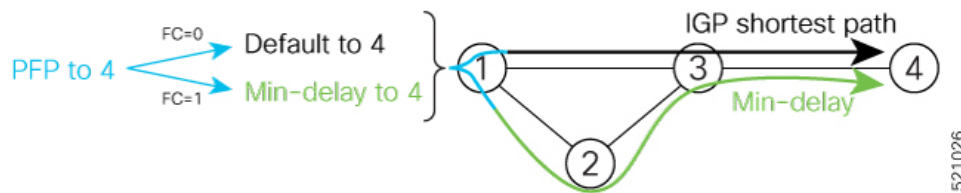
A PFP provides up to 8 "ways" or options to the endpoint. With a PFP, packets are classified by a classification policy and marked using internal tags called forward classes (FCs). The FC setting of the packet selects the "way". For example, this "way" can be a traffic-engineered SR path, using a low-delay 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:

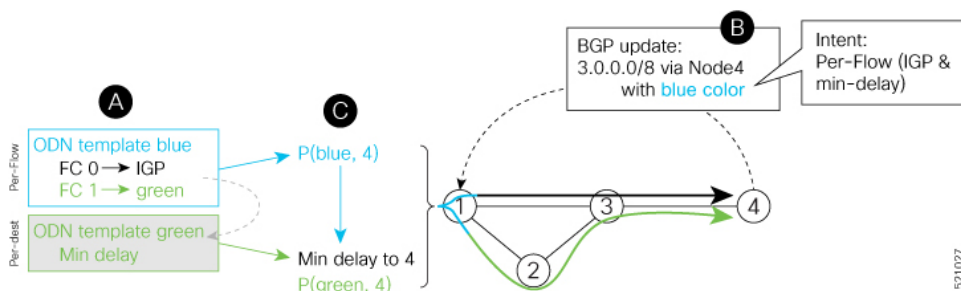
Figure 12: PFP Example



- 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 13: 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 EVPN over PFP is not supported
- Pseudowire and VPLS over PFP are not supported
- BGP PIC is not 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.
- 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 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

```

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

```

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

```

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.

Explicit Binding SID

Use the **binding-sid mpls label** command in SR-TE policy configuration mode to specify the explicit BSID. Explicit BSIDs are allocated from the segment routing local block (SRLB) or the dynamic range of labels. A best-effort is made to request and obtain the BSID for the SR-TE policy. If requested BSID is not available (if it does not fall within the available SRLB or is already used by another application or SR-TE policy), the policy stays down.

Use the **binding-sid explicit {fallback-dynamic | enforce-srlb}** command to specify how the BSID allocation behaves if the BSID value is not available.

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

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

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

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

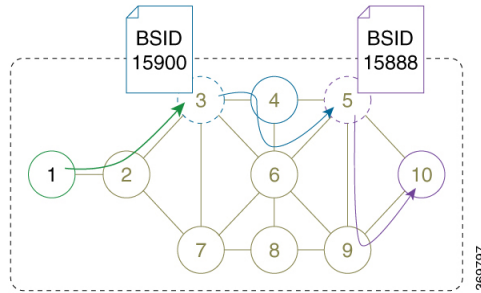
This example shows how to configure an SR policy to use an explicit BSID of 1000. If the BSID is not available, the BSID is allocated dynamically and the policy comes up.

```
segment-routing
traffic-eng
  binding-sid explicit fallback-dynamic
policy goo
  binding-sid mpls 1000
!
```

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 14: Stitching SR-TE Policies Using Binding SID

**Step 1**

On node 5, do the following:

- Define an SR-TE policy with an explicit path configured using the loopback interface IP addresses of node 9 and node 10.
- 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.

Refer to the [EVPN VPWS Preferred Path over SR-TE Policy](#) and [L2VPN VPLS or VPWS Preferred Path over SR-TE Policy](#) sections in the "L2VPN Services over Segment Routing for Traffic Engineering Policy" chapter of the *L2VPN and Ethernet Services Configuration Guide*.

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 6000 Series Routers* *MPLS 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 6000 Series Routers* *MPLS 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

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)

```

Configure Seamless Bidirectional Forwarding Detection

Bidirectional forwarding detection (BFD) provides low-overhead, short-duration detection of failures in the path between adjacent forwarding engines. BFD allows a single mechanism to be used for failure detection

over any media and at any protocol layer, with a wide range of detection times and overhead. The fast detection of failures provides immediate reaction to failure in the event of a failed link or neighbor.

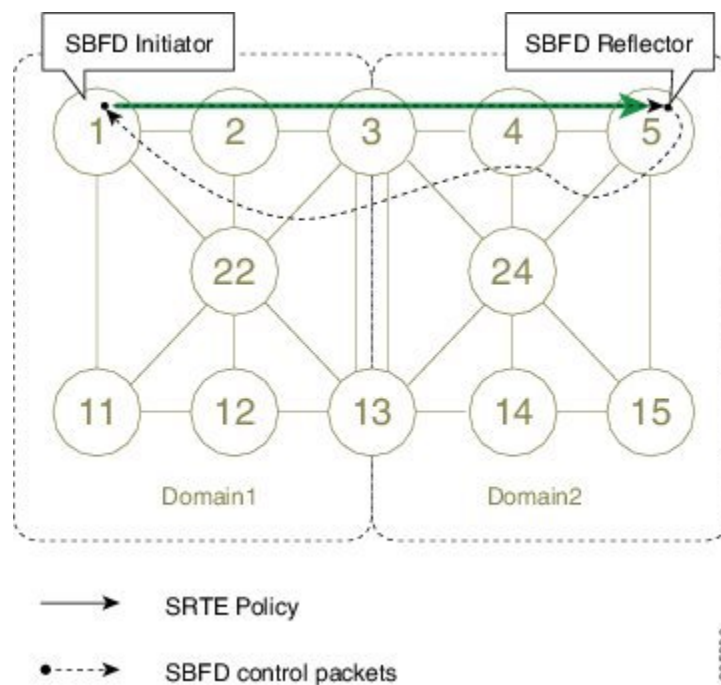
In BFD, each end of the connection maintains a BFD state and transmits packets periodically over a forwarding path. Seamless BFD (SBFD) is unidirectional, resulting in faster session activation than BFD. The BFD state and client context is maintained on the head-end (initiator) only. The tail-end (reflector) validates the BFD packet and responds, so there is no need to maintain the BFD state on the tail-end.

Initiators and Reflectors

SBFD runs in an asymmetric behavior, using initiators and reflectors.

The following figure represents the roles of the SBFD initiator and reflector.

Figure 15: SBFD Initiator and Reflector



The initiator is an SBFD session on a network node that performs a continuity test to a remote entity by sending SBFD packets. The initiator injects the SBFD packets into the segment-routing traffic-engineering (SRTE) policy. The initiator triggers the SBFD session and maintains the BFD state and client context.

The reflector is an SBFD session on a network node that listens for incoming SBFD control packets to local entities and generates response SBFD control packets. The reflector is stateless and only reflects the SBFD packets back to the initiator.

A node can be both an initiator and a reflector, if you want to configure different SBFD sessions.

For SR-TE, SBFD control packets are label switched in forward and reverse direction. For SBFD, the tail-end node is the reflector node; other nodes cannot be a reflector. When using SBFD with SR-TE, if the forward and return directions are label-switched paths, SBFD need not be configured on the reflector node.

Discriminators

The BFD control packet carries 32-bit discriminators (local and remote) to demultiplex BFD sessions. SBFDF requires globally unique SBFDF discriminators that are known by the initiator.

The SBFDF control packets contain the discriminator of the initiator, which is created dynamically, and the discriminator of the reflector, which is configured as a local discriminator on the reflector.

Configure the SBFDF Reflector

To ensure the SBFDF packet arrives on the intended reflector, each reflector has at least one globally unique discriminator. Globally unique discriminators of the reflector are known by the initiator before the session starts. An SBFDF reflector only accepts BFD control packets where "Your Discriminator" is the reflector discriminator.

This task explains how to configure local discriminators on the reflector.

Before you begin

Enable mpls oam on the reflector to install a routing information base (RIB) entry for 127.0.0.0/8.

```
Router_5# configure
Router_5(config)# mpls oam
Router_5(config-oam)#
```

SUMMARY STEPS

1. **configure**
2. **sbfd**
3. **local-discriminator** { *ipv4-address* | *32-bit-value* | **dynamic** | **interface interface** }
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	sbfd Example: Router_5(config)# sbfd	Enters SBFDF configuration mode.
Step 3	local-discriminator { <i>ipv4-address</i> <i>32-bit-value</i> dynamic interface interface } Example: Router_5(config-sbfd)# local-discriminator 10.1.1.5	Configures the local discriminator. You can configure multiple local discriminators.

	Command or Action	Purpose
	<pre>Router_5(config-sbfd) # local-discriminator 987654321 Router_5(config-sbfd) # local-discriminator dynamic Router_5(config-sbfd) # local-discriminator interface Loopback0</pre>	
Step 4	commit	

Verify the local discriminator configuration.

Example

```
Router_5# show bfd target-identifier local

Local Target Identifier Table
-----
Discr          Discr Src   VRF      Status  Flags
                Name
-----
16843013      Local     default  enable  -----ia-
987654321     Local     default  enable  ----v---
2147483649    Local     default  enable  -----d

Legend: TID - Target Identifier
        a - IP Address mode
        d - Dynamic mode
        i - Interface mode
        v - Explicit Value mode
```

What to do next

Configure the SBFDF initiator.

Configure the SBFDF Initiator

Perform the following configurations on the SBFDF initiator.

Enable Line Cards to Host BFD Sessions

The SBFDF initiator sessions are hosted by the line card CPU.

This task explains how to enable line cards to host BFD sessions.

SUMMARY STEPS

1. **configure**
2. **bfd**
3. **multipath include location *node-id***

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	bfd Example: Router_1(config)# bfd	Enters BFD configuration mode.
Step 3	multipath include location <i>node-id</i> Example: Router_1(config-bfd)# multipath include location 0/1/CPU0 Router_1(config-bfd)# multipath include location 0/2/CPU0 Router_1(config-bfd)# multipath include location 0/3/CPU0	Configures BFD multiple path on specific line card. Any of the configured line cards can be instructed to host a BFD session.

What to do next

Map a destination address to a remote discriminator.

Map a Destination Address to a Remote Discriminator

The Sbfd initiator uses a Remote Target Identifier (RTI) table to map a destination address (Target ID) to a remote discriminator.

This task explains how to map a destination address to a remote discriminator.

SUMMARY STEPS

1. **configure**
2. **sbfd**
3. **remote-target ipv4 *ipv4-address***
4. **remote-discriminator *remote-discriminator***

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.

	Command or Action	Purpose
Step 2	sbfd Example: Router_1(config)# sbfd	Enters Sbfd configuration mode.
Step 3	remote-target ipv4 ipv4-address Example: Router_1(config-sbfd)# remote-target ipv4 10.1.1.5	Configures the remote target.
Step 4	remote-discriminator remote-discriminator Example: Router_1(config-sbfd-nnnn)# remote-discriminator 16843013	Maps the destination address (Target ID) to a remote discriminator.

Verify the remote discriminator configuration.

Example

```
Router_1# show bfd target-identifier remote

Remote Target Identifier Table
-----
Discr      Discr Src  VRF      TID Type  Status
-----  -
16843013   Remote    default  ipv4      enable
          10.1.1.5

Legend: TID - Target Identifier
```

What to do next

Enable Sbfd on an SR-TE policy.

Enable Seamless BFD Under an SR-TE Policy or SR-ODN Color Template

This example shows how to enable Sbfd on an SR-TE policy or an SR on-demand (SR-ODN) color template.



Note Do not use BFD with disjoint paths. The reverse path might not be disjoint, causing a single link failure to bring down BFD sessions on both the disjoint paths.

Enable BFD

- Use the **bfd** command in SR-TE policy configuration mode to enable BFD and enters BFD configuration mode.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# bfd
Router(config-sr-te-policy-bfd)#
```

Use the **bfd** command in SR-ODN configuration mode to enable BFD and enters BFD configuration mode.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color 10
Router(config-sr-te-color)# bfd
Router(config-sr-te-color-bfd)#
```

Configure BFD Options

- Use the **minimum-interval** *milliseconds* command to set the interval between sending BFD hello packets to the neighbor. The range is from 15 to 200. The default is 15.

```
Router(config-sr-te-policy-bfd)# minimum-interval 50
```

- Use the **multiplier** *multiplier* command to set the number of times a packet is missed before BFD declares the neighbor down. The range is from 2 to 10. The default is 3.

```
Router(config-sr-te-policy-bfd)# multiplier 2
```

- Use the **invalidation-action** {**down** | **none**} command to set the action to be taken when BFD session is invalidated.

- **down**: LSP can only be operationally up if the BFD session is up
- **none**: BFD session state does not affect LSP state, use for diagnostic purposes

```
Router(config-sr-te-policy-bfd)# invalidation-action down
```

- **(SR-TE policy only)** Use the **reverse-path binding-label** *label* command to specify BFD packets return to head-end by using a binding label.

By default, the S-BFD return path (from tail-end to head-end) is via IPv4. You can use a reverse binding label so that the packet arrives at the tail-end with the reverse binding label as the top label. This label is meant to point to a policy that will take the BFD packets back to the head-end. The reverse binding label is configured per-policy.

Note that when MPLS return path is used, BFD uses echo mode packets, which means the tail-end's BFD reflector does not process BFD packets at all.

The MPLS label value at the tail-end and the head-end must be synchronized by the operator or controller. Because the tail-end binding label should remain constant, configure it as an explicit BSID, rather than dynamically allocated.

```
Router(config-sr-te-policy-bfd)# reverse-path binding-label 24036
```


- Use the **logging session-state-change** command to log when the state of the session changes

```
Router(config-sr-te-policy-bfd) # logging session-state-change
```

Examples

This example shows how to enable SBFD on an SR-TE policy.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# bfd
Router(config-sr-te-policy-bfd) # invalidation-action down
Router(config-sr-te-policy-bfd) # minimum-interval 50
Router(config-sr-te-policy-bfd) # multiplier 2
Router(config-sr-te-policy-bfd) # reverse-path binding-label 24036
Router(config-sr-te-policy-bfd) # logging session-state-change
```

```
segment-routing
 traffic-eng
  policy POLICY1
    bfd
      minimum-interval 50
      multiplier 2
      invalidation-action down
      reverse-path
        binding-label 24036
      !
      logging
        session-state-change
      !
    !
  !
!
```

This example shows how to enable SBFD on an SR-ODN color.

```
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color 10
Router(config-sr-te-color)# bfd
Router(config-sr-te-color-bfd) # minimum-interval 50
Router(config-sr-te-color-bfd) # multiplier 2
Router(config-sr-te-color-bfd) # logging session-state-change
Router(config-sr-te-color-bfd) # invalidation-action down
```

```
segment-routing
 traffic-eng
  on-demand color 10
    bfd
      minimum-interval 50
      multiplier 2
      invalidation-action down
      logging
        session-state-change
      !
    !
  !
!
```

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
  !
!
```




CHAPTER 9

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 extension to support Segment Routing Flexible Algorithm on an MPLS data-plane.

- [Prerequisites for Flexible Algorithm, on page 131](#)
- [Building Blocks of Segment Routing Flexible Algorithm, on page 131](#)
- [Configuring Flexible Algorithm, on page 133](#)
- [Example: Configuring IS-IS Flexible Algorithm, on page 135](#)
- [Example: Traffic Steering to Flexible Algorithm Paths, on page 135](#)

Prerequisites for Flexible Algorithm

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

Building Blocks of Segment Routing Flexible Algorithm

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

Flexible Algorithm Definition

Many possible 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

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

Flexible Algorithm Prefix-SID Advertisement

To be able to forward traffic on a Flexible Algorithm specific path, all routers participating in the Flexible Algorithm will install 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

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](#).

- 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.

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 181](#).

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 175](#).

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.

Configuring Flexible Algorithm

The following IS-IS 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:

- `metric-type delay`



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

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

name—name of the affinity map

- `priority priority value`

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

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

```
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

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: 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

```



```

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
```




CHAPTER 10

Configure Segment Routing Path Computation Element

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

- [About SR-PCE, on page 141](#)
- [Usage Guidelines and Limitations, on page 142](#)
- [Configure SR-PCE, on page 142](#)
- [PCE-Initiated SR Policies, on page 145](#)

About SR-PCE

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



Note For more information on PCE, PCC, and PCEP, refer to the [Path Computation Element](#) section in the *MPLS Configuration Guide for Cisco NCS 6000 Series Routers*.

SR-PCE learns topology information by way of IGP (OSPF or IS-IS) or through BGP Link-State (BGP-LS).

SR-PCE is capable of computing paths using the following methods:

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

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

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

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.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	pce Example:	Enables PCE and enters PCE configuration mode.

	Command or Action	Purpose
	RP/0/RP0/CPU0:router (config) # pce	
Step 3	address ipv4 <i>address</i> Example: RP/0/RP0/CPU0:router (config-pce) # address ipv4 192.168.0.1	Configures a PCE IPv4 address.
Step 4	state-sync ipv4 <i>address</i> Example: 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 <i>size</i> Example: 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 } <i>password</i> Example: 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.
Step 7	segment-routing { strict-sid-only te-latency } Example: RP/0/RP0/CPU0:router (config-pce) # segment-routing strict-sid-only	Configures the segment routing algorithm to use strict SID or TE latency. Note This setting is global and applies to all LSPs that request a path from this controller.
Step 8	timers Example: RP/0/RP0/CPU0:router (config-pce) # timers	Enters timer configuration mode.
Step 9	keepalive <i>time</i> Example: 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.

	Command or Action	Purpose
Step 10	minimum-peer-keepalive <i>time</i> Example: <pre>RP/0/RP0/CPU0:router(config-pce-timers)# minimum-peer-keepalive 30</pre>	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 11	reoptimization <i>time</i> Example: <pre>RP/0/RP0/CPU0:router(config-pce-timers)# reoptimization 600</pre>	Configures the re-optimization timer. The default timer is 1800 seconds.
Step 12	exit Example: <pre>RP/0/RP0/CPU0:router(config-pce-timers)# exit</pre>	Exits timer configuration mode and returns to PCE configuration mode.

Configure the Disjoint Policy (Optional)

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	disjoint-path Example: <pre>RP/0/RP0/CPU0:router(config-pce)# disjoint-path</pre>	Enters disjoint configuration mode.
Step 2	group-id <i>value</i> type {link node srlg srlg-node} [sub-id <i>value</i>] Example:	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):

	Command or Action	Purpose
	<pre>RP/0/RP0/CPU0:router(config-pce-disjoint)# group-id 1 type node sub-id 1</pre>	<ul style="list-style-type: none"> • 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. <p>If a pair of paths that meet the requested disjointness level cannot be found, then the paths will automatically fallback to a lower level:</p> <ul style="list-style-type: none"> • If the requested disjointness level is SRLG or node, then link-disjoint paths will be computed. • If the requested disjointness level was link, or if the first fallback from SRLG or node disjointness failed, then the lists of segments encoding two shortest paths, without any disjointness constraint, will be computed.
Step 3	<p>strict</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-pce-disjoint)# strict</pre>	<p>(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.</p>
Step 4	<p>lsp {1 2} pcc ipv4 address lsp-name lsp_name [shortest-path]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-pce-disjoint)# lsp 1 pcc ipv4 192.168.0.1 lsp-name rtrA_t1 shortest-path RP/0/RP0/CPU0:router(config-pce-disjoint)# lsp 2 pcc ipv4 192.168.0.5 lsp-name rtrE_t2</pre>	<p>Adds LSPs to the disjoint group.</p> <p>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.</p>

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 59](#).

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 selects either the protected or the unprotected Adj-SID of the link, depending on the order in which the Adj-SIDs were received.

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
  !
!
```




CHAPTER 11

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.

Table 5: 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. Delay profile allows you to schedule probe and configure metric advertisement parameters for delay measurement.
Protocols	Two-Way Active Measurement Protocol (TWAMP) Light (using RFC 5357 with IP/UDP encap).
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.

The following are the means by which you can measure the performance of your network:

- [Liveness Monitoring, on page 150](#)
- [Delay Measurement, on page 150](#)

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.

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.

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 measurement 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.

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 153](#) Use to monitor delay experienced by data packets in a single link or path between two nodes in a network.
- [Delay Measurement for IP Endpoint:](#) 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.

Measurement Modes

The following table compares the different hardware and timing requirements for the measurement modes supported in SR PM.

Table 6: Measurement Mode Requirements

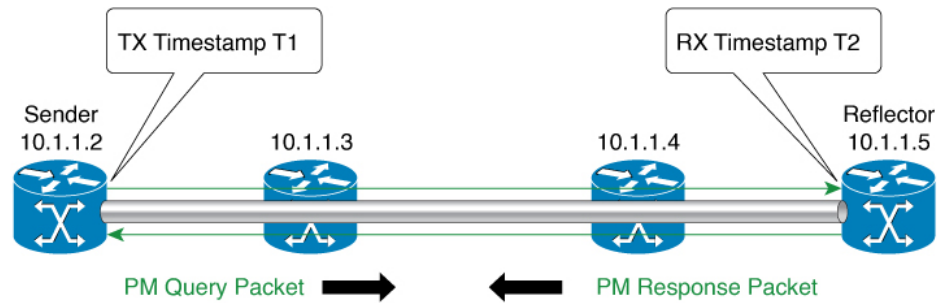
Measurement Mode	Sender: PTP-Capable HW and HW Timestamping	Reflector: PTP-Capable HW and HW Timestamping	PTP Clock Synchronization between Sender and Reflector
One-way	Required	Required	Required
Two-way	Required	Required	Not Required

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 16: One-Way



- One Way Delay = $(T2 - T1)$
- Hardware clock synchronized using PTP (IEEE 1588) between sender and reflector nodes (all nodes for higher accuracy)

521501

The PM query and response for one-way delay measurement can be described in the following steps:

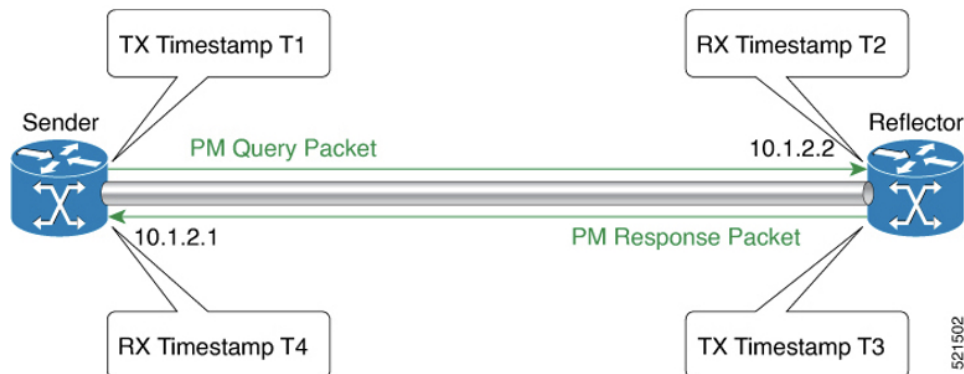
1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. The ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets 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 measurement 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 17: Two-Way



521502

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.

Link Delay Measurement

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
Segment Routing Performance Measurement for Link Delay Measurement on Link Bundles	Release 7.2.2	<p>This enhancement introduces support for using link bundles with Segment Routing Performance Measurement link delay measurement</p> <p>A link bundle is a group of one or more ports that are aggregated together and treated as a single link. Link bundles allow you to group multiple point-to-point links together into one logical link and provide higher bidirectional bandwidth, redundancy, and load balancing between two routers.</p>

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.

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 151](#) for more information.
- **protocol:** Interface delay measurement using RFC 5357 with IP/UDP encapsulation (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)# 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.

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

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
```

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 [<i>interface</i>]	Displays the PM link-delay probe history for interfaces.
show performance-measurement history aggregated interfaces [<i>interface</i>]	Displays the PM link-delay aggregated history for interfaces.
show performance-measurement history advertisement interfaces [<i>interface</i>]	Displays the PM link-delay advertisement history for interfaces.
show performance-measurement counters [<i>interface interface</i>] [<i>location location-name</i>]	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 [<i>location location-name</i>]	Displays the PM for link-delay summary on the remote-end router (responder).
show performance-measurement responder interfaces [<i>interface</i>]	Displays PM for link-delay for interfaces on the remote-end router.
show performance-measurement responder counters [<i>interface interface</i>] [<i>location location-name</i>]	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 Measurement for IP Endpoint

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
IP Endpoint Delay Measurement Monitoring		<p>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).</p> <p>This feature is supported on IPv4, IPv6, and MPLS data planes.</p>

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.

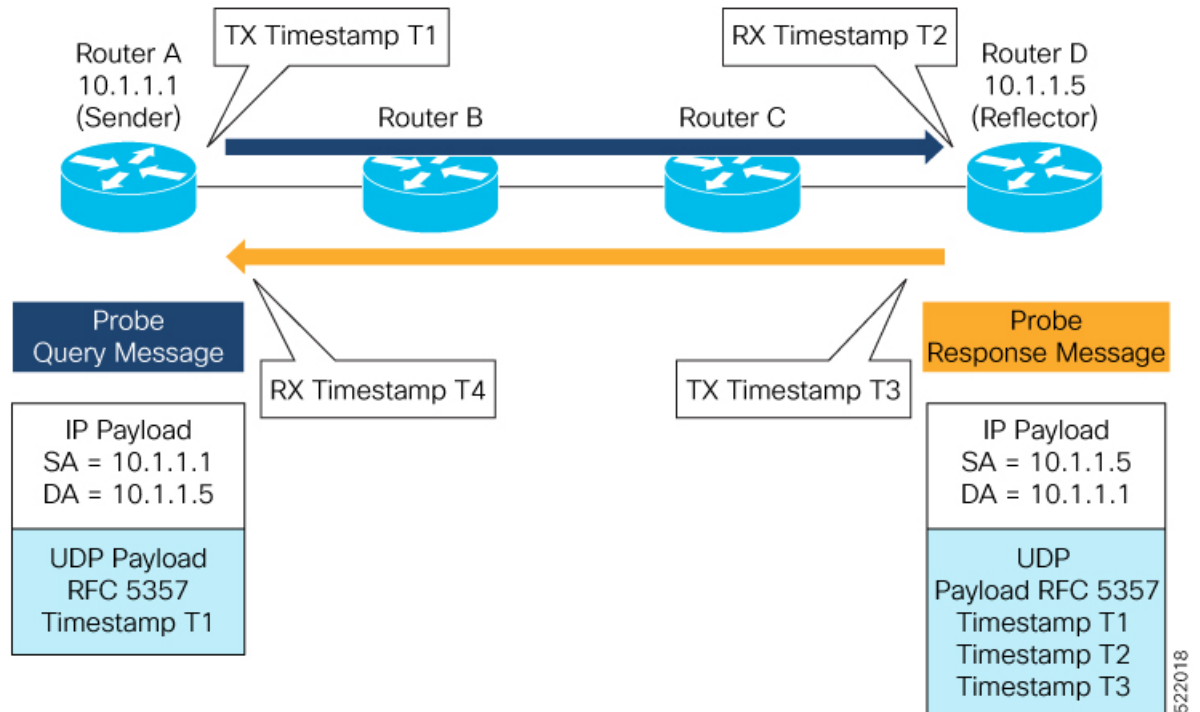
IP Endpoint Delay Measurement over MPLS Network Usecases

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 18: 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 occurred

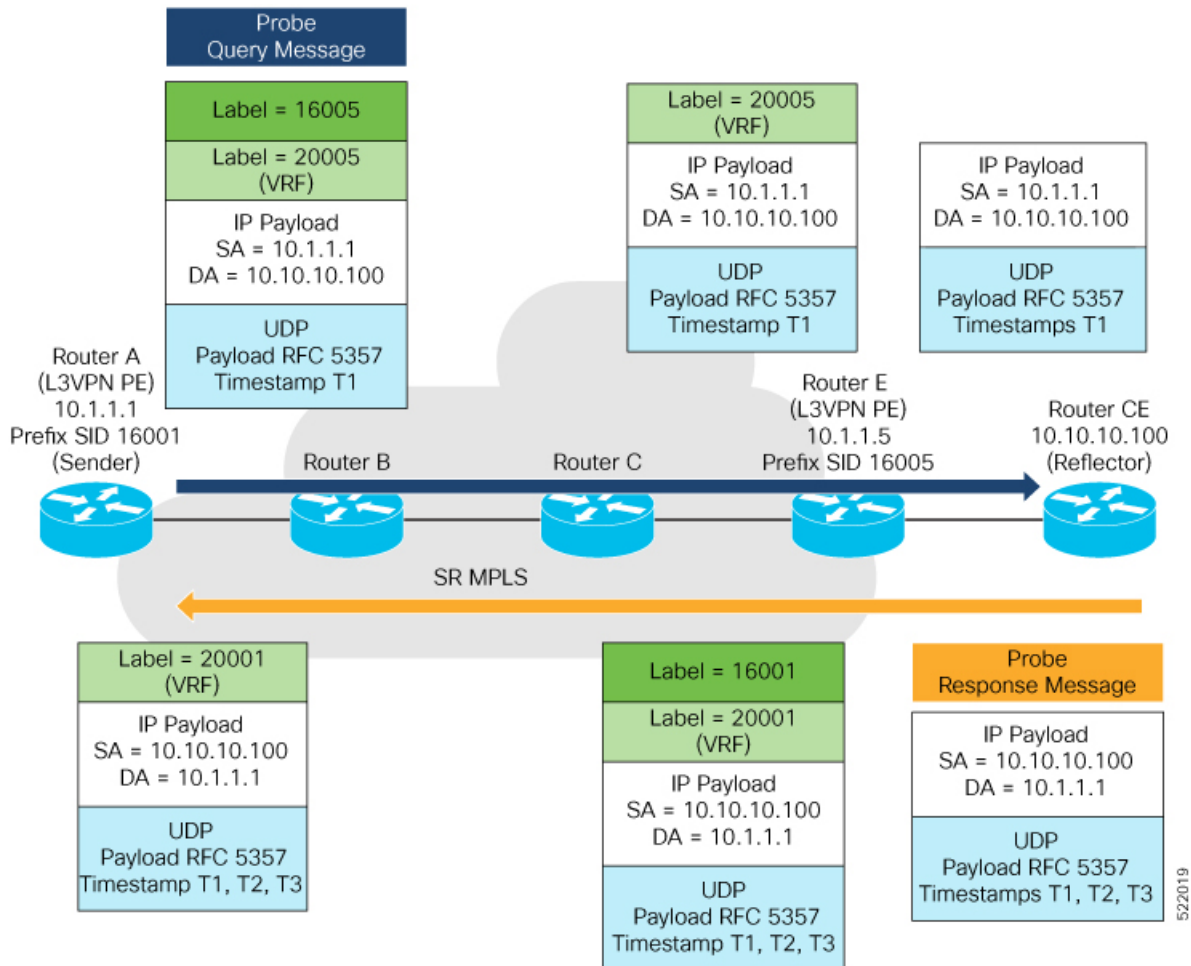
  Next advertisement:
    Threshold check scheduled in 4 more probes (roughly every 120 seconds)
    No probes completed

  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 19: 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 occurred

Next advertisement:
  Advertisement not scheduled as the probe is not running

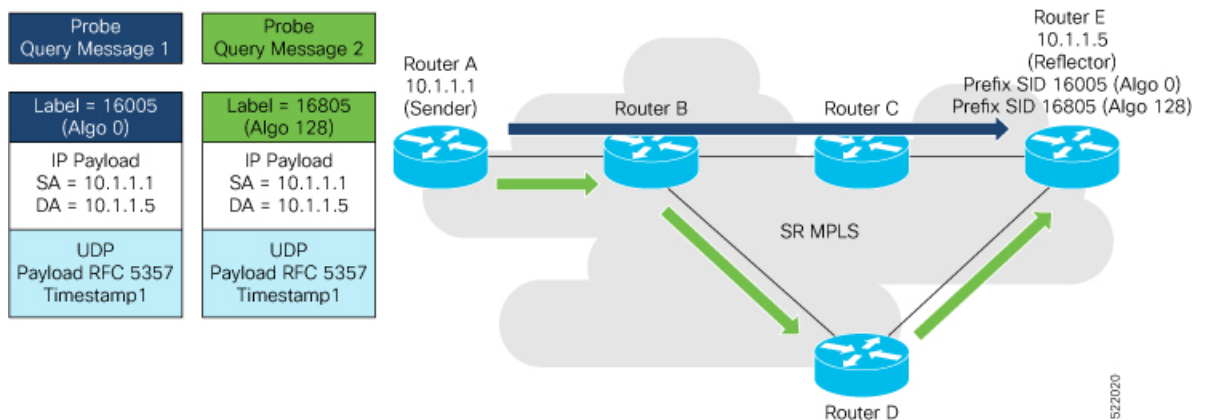
Current computation:
  Not running: Unable to resolve (non-existing) vrf
```

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 20: 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 occurred
```

```
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 occurred
```

```
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 occurred
```

```
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.

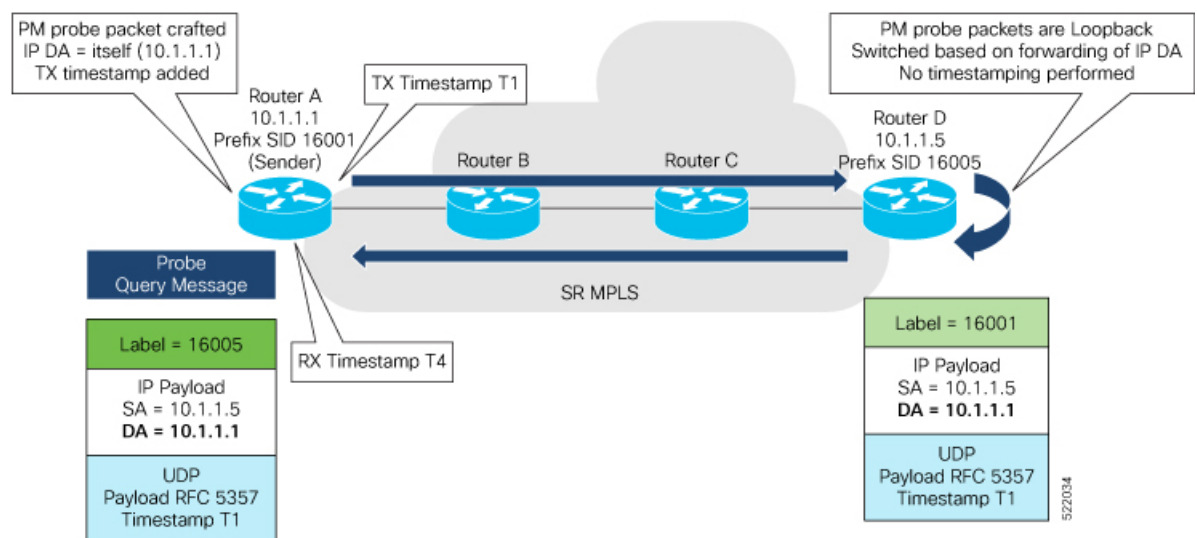
The received timestamp (T4) stored.

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 21: 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

```




CHAPTER 12

Configure Topology-Independent Loop-Free Alternate (TI-LFA)

Topology-Independent Loop-Free Alternate (TI-LFA) uses segment routing to provide link 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.

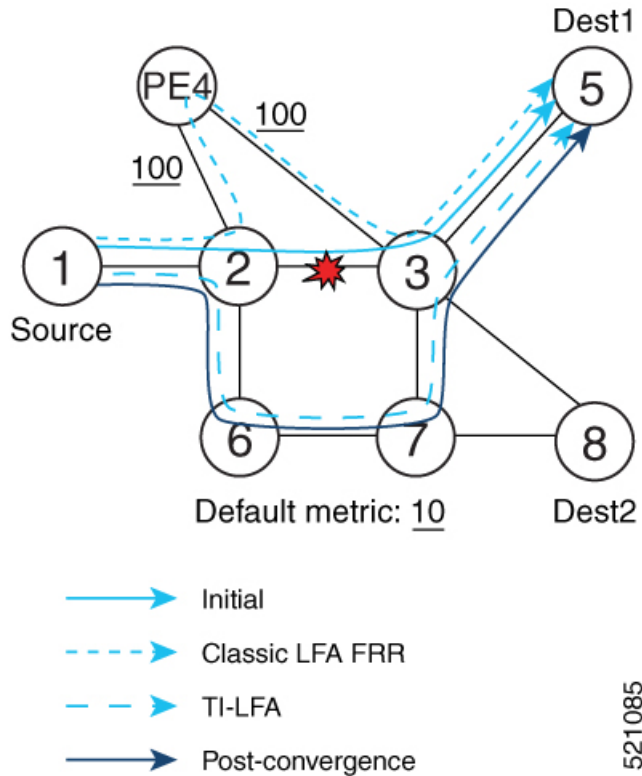
The goal of TI-LFA is to reduce the packet loss that results while routers converge after a topology change due to a link failure. Rapid failure repair (< 50 msec) is achieved through the use of pre-calculated backup paths that are loop-free and safe to use until the distributed network convergence process 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 will be used when other links fail.
- Simple to operate — There is no need to perform a case-by-case adjustment 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 22: 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.

- [Usage Guidelines and Limitations, on page 176](#)
- [Configuring TI-LFA for IS-IS, on page 177](#)
- [Configuring TI-LFA for OSPF, on page 179](#)

Usage Guidelines and Limitations

The TI-LFA guidelines and limitations are listed below:

TI-LFA Functionality	IS-IS ¹	OSPFv2
<i>Protected Traffic Types</i>		

TI-LFA Functionality	IS-IS ¹	OSPFv2
Protection for SR labeled traffic	Supported	Supported
Protection of IPv4 unlabeled traffic	Supported (IS-ISv4)	Supported
Protection of IPv6 unlabeled traffic	Unsupported	N/A
Protection Types		
Link Protection	Supported	Supported
Node Protection	Supported	Supported
Local SRLG Protection	Supported	Supported
Weighted Remote SRLG Protection		
Line Card Disjoint Protection	Supported	Unsupported
Interface Types		
Ethernet Interfaces	Supported	Supported
Ethernet Bundle Interfaces	Supported	Supported
TI-LFA over GRE Tunnel as Protecting Interface		
Additional Functionality		
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 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 29](#).

SUMMARY STEPS

1. **configure**
2. **router isis *instance-id***
3. **interface *type interface-path-id***
4. **address-family ipv4 [unicast]**
5. **fast-reroute per-prefix**
6. **fast-reroute per-prefix ti-lfa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters XR Config mode.
Step 2	router isis <i>instance-id</i> Example: 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. 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 <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router (config-isis)# interface GigabitEthernet0/0/2/1	Enters interface configuration mode.
Step 4	address-family ipv4 [unicast] Example: 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 Example: RP/0/RP0/CPU0:router (config-isis-if-af)# fast-reroute per-prefix	Enables per-prefix fast reroute.
Step 6	fast-reroute per-prefix ti-lfa Example: RP/0/RP0/CPU0:router (config-isis-if-af)# fast-reroute per-prefix ti-lfa	Enables per-prefix TI-LFA fast reroute link protection.

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 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 41](#).

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **fast-reroute per-prefix**
6. **fast-reroute per-prefix ti-lfa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/RP0/CPU0:router(config)# <code>router ospf 1</code>	Enables OSPF routing for the specified routing process, and places the router in router configuration mode.
Step 3	area <i>area-id</i> Example: RP/0/RP0/CPU0:router(config-ospf)# <code>area 1</code>	Enters area configuration mode.

	Command or Action	Purpose
Step 4	interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-ospf-ar) # interface GigabitEthernet0/0/2/1	Enters interface configuration mode.
Step 5	fast-reroute per-prefix Example: 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 Example: RP/0/RP0/CPU0:router(config-ospf-ar-if) # fast-reroute per-prefix ti-lfa	Enables per-prefix TI-LFA fast reroute link protection.

TI-LFA has been successfully configured for segment routing.



CHAPTER 13

Configure Segment Routing Microloop Avoidance

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

- [About Segment Routing Microloop Avoidance, on page 181](#)
- [Configure Segment Routing Microloop Avoidance for IS-IS, on page 183](#)
- [Configure Segment Routing Microloop Avoidance for OSPF, on page 184](#)

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.

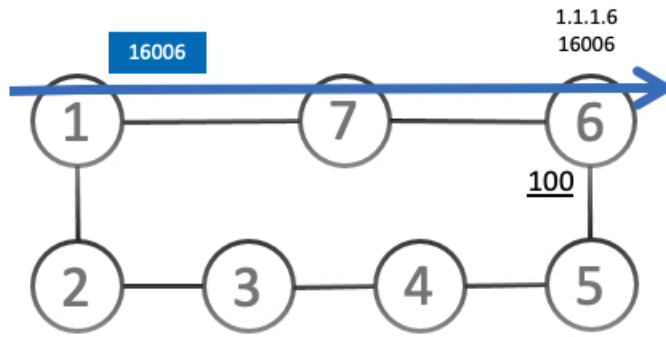
If a node determines that a microloop could occur on the new topology, the IGP computes a microloop-avoidant path by updating the forwarding table and temporarily (based on a RIB update delay timer) installing the SID-list imposition entries associated with the microloop-avoidant path for the destination. 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.

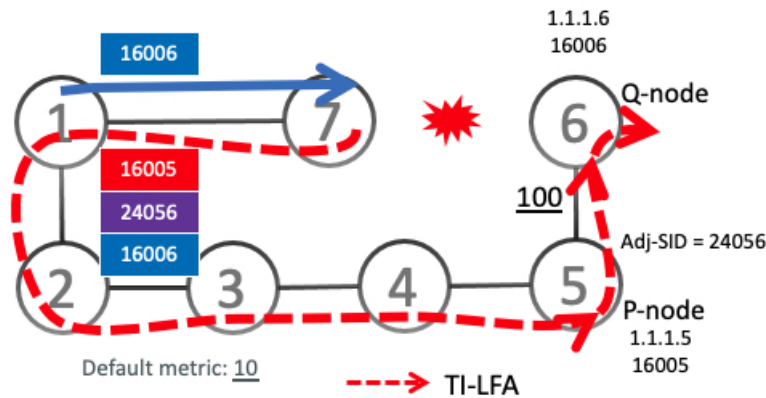
SR microloop avoidance is a local behavior and therefore not all nodes need to implement it to get the benefits.

In the topology below, microloops can occur after the failure of the link between Node6 and Node7.

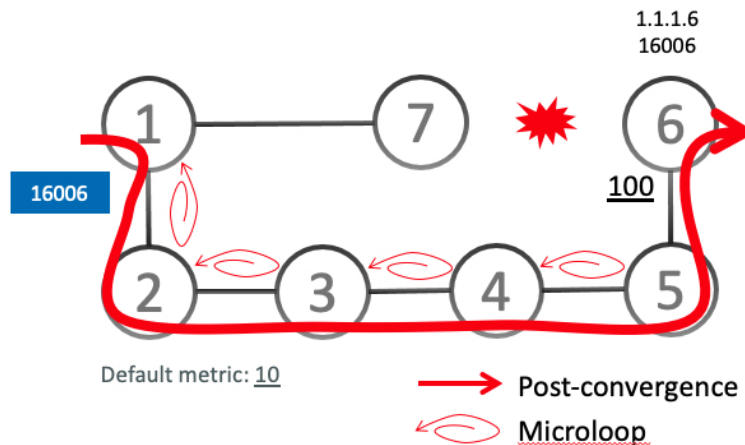
At steady state, Node1 sends traffic to node 6 (16006) via Node7. Node 7 is configured with TI-LFA to protect traffic to Node6.



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.



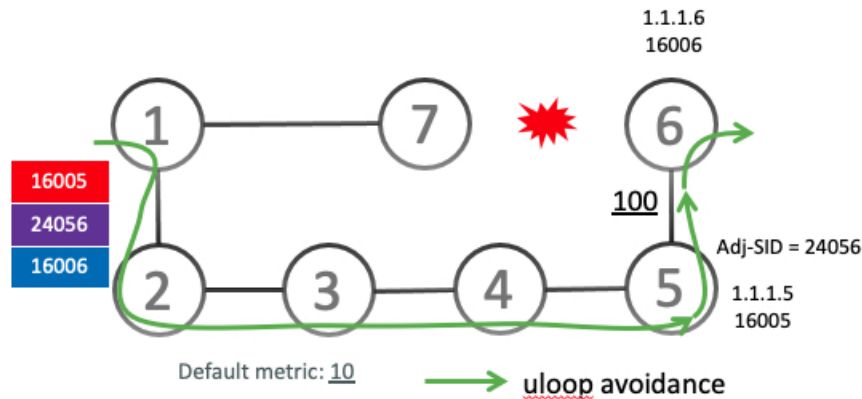
However, if nodes along the path do not converge at the same time, microloops can be introduced. For example, if Node2 converged before Node3, Node3 would send traffic back to Node2 as the shortest IGP path to Node6. The traffic between Node2 and Node3 creates a microloop.



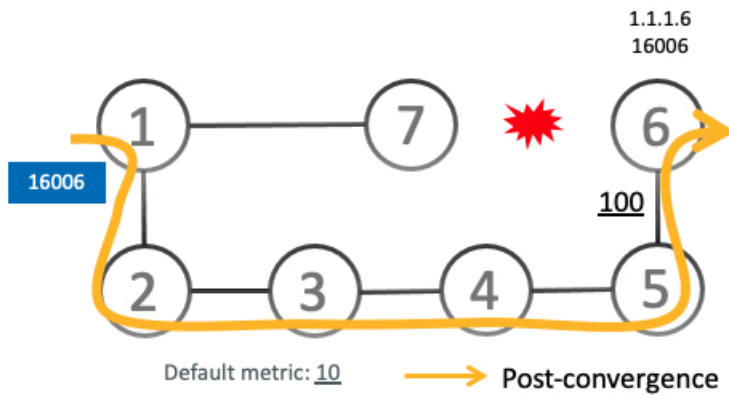
With microloop avoidance configured on Node1, a post-convergence path is computed and possible microloops on the post-convergence path for any destination are detected.

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.



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 29](#).

Procedure

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	router isis <i>instance-id</i> Example: RP/0/RP0/CPU0:router(config)# <code>router isis 1</code>	Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode. You can change the level of routing to be performed by a particular routing instance by using the is-type router configuration command.
Step 3	address-family ipv4 [unicast] Example: RP/0/RP0/CPU0:router(config-isis)# <code>address-family ipv4 unicast</code>	Specifies the IPv4 address family and enters router address family configuration mode.
Step 4	microloop avoidance segment-routing Example: RP/0/RP0/CPU0:router(config-isis-af)# <code>microloop avoidance segment-routing</code>	Enables Segment Routing Microloop Avoidance.
Step 5	microloop avoidance rib-update-delay <i>delay-time</i> Example: RP/0/RP0/CPU0:router(config-isis-af)# <code>microloop avoidance rib-update-delay 3000</code>	Specifies the amount of time the node uses the microloop avoidance policy before updating its forwarding table. The <i>delay-time</i> 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 41](#).

Procedure

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/RP0/CPU0:router(config)# <code>router ospf 1</code>	Enables OSPF routing for the specified routing process, and places the router in router configuration mode.
Step 3	microloop avoidance segment-routing Example: RP/0/RP0/CPU0:router(config-ospf)# <code>microloop avoidance segment-routing</code>	Enables Segment Routing Microloop Avoidance.
Step 4	microloop avoidance rib-update-delay <i>delay-time</i> Example: RP/0/RP0/CPU0:router(config-ospf)# <code>microloop avoidance rib-update-delay 3000</code>	Specifies the amount of time the node uses the microloop avoidance path before updating its forwarding table. The <i>delay-time</i> is in milliseconds. The range is from 1-60000. The default value is 5000.



CHAPTER 14

Configure Segment Routing Mapping Server

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

- [Segment Routing Mapping Server, on page 187](#)
- [Segment Routing and LDP Interoperability, on page 188](#)
- [Configuring Mapping Server, on page 191](#)
- [Enable Mapping Advertisement, on page 193](#)
- [Enable Mapping Client, on page 195](#)

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

- The position of the mapping server in the network is not important. However, since the mapping advertisements are distributed in IGP using the regular IGP advertisement mechanism, the mapping server needs an IGP adjacency to the network.
- The role of the mapping server is crucial. For redundancy purposes, you should configure multiple mapping servers in the networks.
- The mapping server functionality does not support a scenario where SID-mapping entries learned through one IS-IS instance are used by another IS-IS instance to determine the prefix-SID of a prefix. For example, mapping entries learnt from remote routers by 'router isis 1' cannot be used to calculate prefix-SIDs for prefixes learnt, advertised, or downloaded to FIB by 'router isis 2'. A mapping server is required for each IS-IS instance.
- Segment Routing Mapping Server does not support Virtual Routing and Forwarding (VRF) currently.

Segment Routing and LDP Interoperability

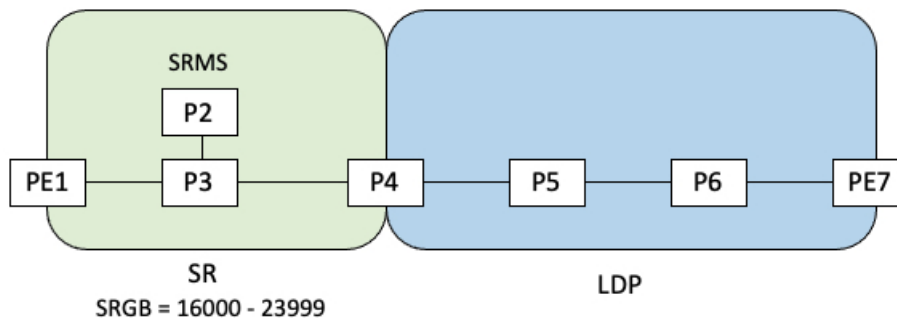
IGP provides mechanisms through which segment routing (SR) interoperates with label distribution protocol (LDP). The control plane of segment routing co-exists with LDP.

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.

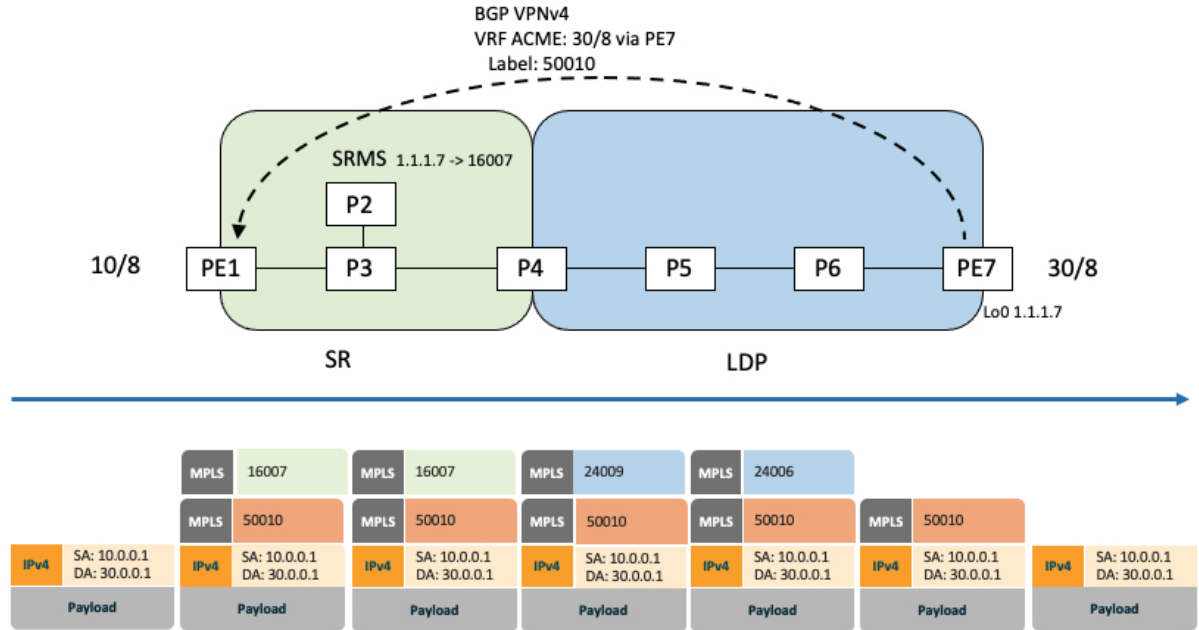
Consider a network as shown in the following figure. This network is a mix of both LDP and SR-capable nodes.



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.

SUMMARY STEPS

1. **configure**
2. **segment-routing**
3. **mapping-server**
4. **prefix-sid-map**
5. **address-family ipv4 |ipv6**
6. *ip-address/prefix-length first-SID-value range range*
7. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	segment-routing Example: RP/0/RP0/CPU0:router(config)# <code>segment-routing</code>	Enables segment routing.
Step 3	mapping-server Example: RP/0/RP0/CPU0:router(config-sr)# <code>mapping-server</code>	Enables mapping server configuration mode.
Step 4	prefix-sid-map Example: RP/0/RP0/CPU0:router(config-sr-ms)# <code>prefix-sid-map</code>	Enables prefix-SID mapping configuration mode. Note Two-way prefix SID can be enabled directly under IS-IS or through a mapping server.
Step 5	address-family ipv4 ipv6 Example: This example shows the address-family for ipv4:	Configures address-family for IS-IS.

	Command or Action	Purpose
	<pre>RP/0/RP0/CPU0:router(config-sr-ms-map)# address-family ipv4</pre> <p>This example shows the address-family for ipv6:</p> <pre>RP/0/RP0/CPU0:router(config-sr-ms-map)# address-family ipv6</pre>	
Step 6	<p><i>ip-address/prefix-length first-SID-value range range</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-sr-ms-map-af)# 10.1.1.1/32 10 range 200 RP/0/RP0/CPU0:router(config-sr-ms-map-af)# 20.1.0.0/16 400 range 300</pre>	<p>Adds SID-mapping entries in the active local mapping policy. In the configured example:</p> <ul style="list-style-type: none"> • Prefix 10.1.1.1/32 is assigned prefix-SID 10, prefix 10.1.1.2/32 is assigned prefix-SID 11,..., prefix 10.1.1.199/32 is assigned prefix-SID 200 • Prefix 20.1.0.0/16 is assigned prefix-SID 400, prefix 20.2.0.0/16 is assigned prefix-SID 401,..., and so on.
Step 7	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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

SUMMARY STEPS

1. **router isis** *instance-id*
2. **address-family** { **ipv4** | **ipv6** } [**unicast**]
3. **segment-routing prefix-sid-map advertise-local**
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	router isis <i>instance-id</i> Example: 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. <ul style="list-style-type: none"> • 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] Example: The following is an example for ipv4 address family: 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 Example:	Configures IS-IS to advertise locally configured prefix-SID mappings.

	Command or Action	Purpose
	RP/0/RP0/CPU0:router(config-isis-af)# segment-routing prefix-sid-map advertise-local	
Step 4	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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

SUMMARY STEPS

1. **router ospf** *process-name*
2. **segment-routing prefix-sid-map advertise-local**
3. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>router ospf <i>process-name</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# router ospf 1</pre>	Enables OSPF routing for the specified routing instance, and places the router in router configuration mode.
Step 2	<p>segment-routing prefix-sid-map advertise-local</p> <p>Example:</p>	Configures OSPF to advertise locally configured prefix-SID mappings.

	Command or Action	Purpose
	RP/0/RP0/CPU0:router (config-ospf) # segment-routing prefix-sid-map advertise-local	
Step 3	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • 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
```



CHAPTER 15

Using Segment Routing Traffic Matrix

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

- [Segment Routing Traffic Matrix, on page 197](#)
- [Traffic Collector Process, on page 197](#)
- [Configuring Traffic Collector, on page 198](#)
- [Displaying Traffic Information, on page 200](#)

Segment Routing Traffic Matrix

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

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

Traffic Collector Process

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

Pcounters

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

- Base Pcounter – any packet that is switched on the prefix-SID forwarding information base (FIB) entry
- TM Pcounter – any packet from an external interface and switched on the prefix-SID FIB entry

The Traffic Collector periodically collects the Base Pcounters and TM Pcounters of all prefix-SIDs, and the Pcounters of all tunnel interfaces.

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

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

Feature Support and Limitations

- Pcounters for IPv4 SR Prefix SIDs are supported.
- Pcounters for IPv6 SR Prefix SIDs are not supported.
- TM Pcounters increment for incoming SR-labeled and IP traffic destined for an SR Prefix SID.
- External interface support can be enabled on all Ethernet interfaces except Management, Bundle, and sub interfaces. Tunnels may not be set as external interfaces.
- Default VRF is supported. Non-default VRF is not supported.

Configuring Traffic Collector

Perform these tasks to configure the traffic collector.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters XR Config mode.
Step 2	traffic-collector Example:	Enables traffic collector and places the router in traffic collector configuration mode.

	Command or Action	Purpose
	<code>RP/0/RP0/CPU0:router(config)# traffic-collector</code>	
Step 3	<p>statistics collection-interval <i>value</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-tc)# statistics collection-interval 5</pre>	<p>(Optional) Sets the frequency that the traffic collector collects and posts data, in minutes. Valid values are 1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, and 60. The default interval is 1.</p>
Step 4	<p>statistics history-size <i>value</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-tc)# statistics history-size 10</pre>	<p>(Optional) Specifies the number of entries kept in the history database. Valid values are from 1 to 10. The default is 5.</p> <p>Note The number of entries affects how the average packet and average byte rates are calculated. The rates are calculated over the range of the histories and are not averages based in real time.</p>
Step 5	<p>statistics history-timeout <i>value</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-tc)# statistics history-timeout 24</pre>	<p>(Optional) When a prefix SID or a tunnel-te interface is deleted, the history-timeout sets the length of time, in hours, that the prefix SID and tunnel statistics are retained in the history before they are removed. The minimum is one hour; the maximum is 720 hours. The default is 48.</p> <p>Note Enter 0 to disable the history timeout. (No history is retained.)</p>
Step 6	<p>interface <i>type l3-interface-address</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-tc)# interface TenGigE 0/1/0/3</pre>	<p>Identifies interfaces that handle external traffic. Only L3 interfaces are supported for external traffic.</p>
Step 7	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

This completes the configuration for the traffic collector.

Displaying Traffic Information

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



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

- Display the configured external interfaces:

```
RP/0/RSP0/CPU0:router# show traffic-collector external-interface
Interface                Status
-----                -
Te0/1/0/3                Enabled
Te0/1/0/4                Enabled
```

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

```
RP/0/RSP0/CPU0:router# show traffic-collector ipv4 counters prefix 10.1.1.10/32 detail
Prefix: 10.1.1.10/32 Label: 16010 State: Active
Base:
  Average over the last 5 collection intervals:
    Packet rate: 9496937 pps, Byte rate: 9363979882 Bps

  History of counters:
    23:01 - 23:02: Packets 9379529, Bytes: 9248215594
    23:00 - 23:01: Packets 9687124, Bytes: 9551504264
    22:59 - 23:00: Packets 9539200, Bytes: 9405651200
    22:58 - 22:59: Packets 9845278, Bytes: 9707444108
    22:57 - 22:58: Packets 9033554, Bytes: 8907084244
TM Counters:
  Average over the last 5 collection intervals:
    Packet rate: 9528754 pps, Byte rate: 9357236821 Bps

  History of counters:
    23:01 - 23:02: Packets 9400815, Bytes: 9231600330
    23:00 - 23:01: Packets 9699455, Bytes: 9524864810
    22:59 - 23:00: Packets 9579889, Bytes: 9407450998
    22:58 - 22:59: Packets 9911734, Bytes: 9733322788
    22:57 - 22:58: Packets 9051879, Bytes: 8888945178
```

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

- Display the counter history database for a policy:

```
RP/0/RSP0/CPU0:router# show traffic-collector counters tunnels srte_c_12_ep_6.6.6.2
detail
Tunnel: srte_c_12_ep_6.6.6.2 State: Active
  Average over the last 5 collection intervals:
    Packet rate: 9694434 pps, Byte rate: 9597489858 Bps

  History of counters:
    23:14 - 23:15: Packets 9870522 , Bytes: 9771816780
```

```

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

```

This output shows the average Pcounter (packets, bytes), the Pcounter history, and the collection interval for the policy.

Configuring Telemetry Session to Export SR-TM Statistics

The following configuration shows how to enable the SR-TM and the corresponding model-driven streaming Telemetry paths for exporting SR-TM statistics. Refer to the *Telemetry Configuration Guide for Cisco NCS 6000 Series Routers* for details on telemetry.

```

/* Configure the SR-TM */

traffic-collector
interface TenGigE0/3/0/0/8
!
statistics
  history-size 10
  history-timeout 1
!
!
/* Configure streaming telemetry */

telemetry model-driven
destination-group test-collector-1
  address-family ipv4 10.1.201.250 port 1624
  encoding self-describing-gpb
  protocol tcp
!
!
destination-group test-collector-2
  address-family ipv4 10.30.110.147 port 31500
  encoding self-describing-gpb
  protocol tcp
!
!

/* Configure telemetry paths for SR-TM */

sensor-group test-sg-SRTM
  sensor-path Cisco-IOS-XR-infra-tc-oper:traffic-collector/summary
  sensor-path Cisco-IOS-XR-infra-tc-oper:traffic-collector/afs/af/counters/tunnels/tunnel
  sensor-path Cisco-IOS-XR-infra-tc-oper:traffic-collector/afs/af/counters/prefixes/prefix
  sensor-path
Cisco-IOS-XR-infra-tc-oper:traffic-collector/external-interfaces/external-interface
  sensor-path
Cisco-IOS-XR-infra-tc-oper:traffic-collector/vrf-table/default-vrf/afs/af/counters/tunnels/tunnel
  sensor-path
Cisco-IOS-XR-infra-tc-oper:traffic-collector/vrf-table/default-vrf/afs/af/counters/prefixes/prefix
!
subscription test-subscription
  sensor-group-id test-sg-SRTM sample-interval 60000
  destination-id test-collector-1
  destination-id test-collector-2
  source-interface Loopback0
!
!

```




CHAPTER 16

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, Nil-FEC (forwarding equivalence classes) LSP Ping and Traceroute functionality.

- [MPLS Ping and Traceroute for BGP and IGP Prefix-SID, on page 203](#)
- [Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID, on page 204](#)
- [MPLS LSP Ping and Traceroute Nil FEC Target, on page 205](#)
- [Examples: LSP Ping and Traceroute for Nil_FEC Target, on page 206](#)
- [Segment Routing Ping and Traceroute, on page 207](#)
- [Segment Routing Data Plane Monitoring, on page 212](#)

MPLS Ping and Traceroute for BGP and IGP Prefix-SID

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

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

The MPLS LSP Ping feature is used to check the connectivity between ingress Label Switch Routers (LSRs) and egress LSRs along an LSP. MPLS LSP ping uses MPLS echo request and reply messages, similar to Internet Control Message Protocol (ICMP) echo request and reply messages, to validate an LSP. The destination IP address of the MPLS echo request packet is different from the address used to select the label stack. The destination IP address is defined as a 127.x.y.z/8 address and it prevents the IP packet from being IP switched to its destination, if the LSP is broken.

The MPLS LSP Traceroute feature is used to isolate the failure point of an LSP. It is used for hop-by-hop fault localization and path tracing. The MPLS LSP Traceroute feature relies on the expiration of the Time to Live (TTL) value of the packet that carries the echo request. When the MPLS echo request message hits a transit node, it checks the TTL value and if it is expired, the packet is passed to the control plane, else the

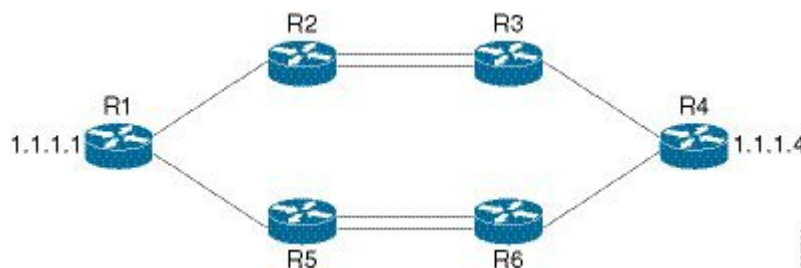
message is forwarded. If the echo message is passed to the control plane, a reply message is generated based on the contents of the request message.

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

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

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

These examples use the following topology:



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

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 9: 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:                  GigabitEthernet0/2/0/1   GigabitEthernet0/2/0/1
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   -----
-----
16004  Pop         No ID       Gi0/2/0/1   10.1.1.4    1392
      Pop         No ID       Gi0/2/0/2   10.1.2.2     0
16005  16005      No ID       Gi0/2/0/0   10.1.1.4     0
      16005      No ID       Gi0/2/0/1   10.1.2.2     0
16007  16007      No ID       Gi0/2/0/0   10.1.1.4    4752
      16007      No ID       Gi0/2/0/1   10.1.2.2     0
24000  Pop         SR Adj (idx 0)  Gi0/2/0/0   10.1.1.4     0
24001  Pop         SR Adj (idx 2)  Gi0/2/0/0   10.1.1.4     0
24002  Pop         SR Adj (idx 0)  Gi0/2/0/1   10.1.2.2     0
24003  Pop         SR Adj (idx 2)  Gi0/2/0/1   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 0/2/0/1 nexthop 10.1.1.4 repeat 1
Sending 1, 72-byte MPLS Echos with Nil FEC labels 16005,16007,
  timeout is 2 seconds, send interval is 0 msec:
```

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

```
Type escape sequence to abort.
```

```
!
Success rate is 100 percent (1/1), round-trip min/avg/max = 1/1/1 ms
Total Time Elapsed 0 ms
```

Traceroute Nil FEC Target

```
RP/0/RP0/CPU0:router-arizona# traceroute mpls nil-fec labels 16005,16007 output interface
GigabitEthernet 0/2/0/1 nexthop 10.1.1.4
Tracing MPLS Label Switched Path with Nil FEC labels 16005,16007, timeout is 2 seconds
```

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

```
Type escape sequence to abort.
```

```
0 10.1.1.3 MRU 1500 [Labels: 16005/16007/explicit-null Exp: 0/0/0]
L 1 10.1.1.4 MRU 1500 [Labels: implicit-null/16007/explicit-null Exp: 0/0/0] 1 ms
L 2 10.1.1.5 MRU 1500 [Labels: implicit-null/explicit-null Exp: 0/0] 1 ms
! 3 10.1.1.7 1 ms
```

Segment Routing Ping and Traceroute

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

```

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

```

Segment Routing Data Plane Monitoring

Unreported traffic drops in MPLS networks could be difficult to detect and isolate. They can be caused by user configuration, out-of-sync neighbors, or incorrect data-plane programming. Segment Routing Data Plane Monitoring (SR DPM) provides a scalable solution to address data-plane consistency verification and detection of unreported traffic drops. SR DPM validates the actual data plane status of all FIB entries associated with SR IGP prefix SIDs.

The primary benefits of SR DPM include:

- **Automation** – A node automatically verifies the integrity of the actual forwarding entries exercised by transit traffic.
- **Comprehensive Coverage** – Tests validate forwarding consistency for each set of destination prefixes across each combination of upstream and downstream neighbors and across all ECMP possibilities.
- **Scalability** – SR DPM is a highly scalable solution due to its localized detection process.
- **Proactive and Reactive modes of operation** – Solution caters to both continuous and on-demand verification.
- **Standards-based** – SR DPM uses existing MPLS OAM tools and leverages SR to enforce test traffic path.

DPM performs data plane validation in two phases:

- **Adjacency Validation**—Using special MPLS echo request packets, adjacency validation ensures that all local links are able to forward and receive MPLS traffic correctly from their neighbors. It also ensures that DPM is able to verify all local adjacency SID labels and to flag any inconsistencies, including traffic drops, forwarding by the local or neighboring device to an incorrect neighbor that is not associated with the specified adjacency, or forwarding by the local or neighboring device to the correct neighbor but over an incorrect link not associated with the specified adjacency. DPM validates the following adjacencies for each link when available:
 - Unprotected adjacency
 - Protected adjacency
 - Static adjacency

- Dynamic adjacency
- Shared adjacency



Note Observe the following limitations for adjacency validation:

- The adjacency validation phase only validates links that are participating in IGP (OSPF and IS-IS) instances. If one or more link is not part of the IGP, it will not be validated since there are no Adjacency SID labels.
 - Adjacency validation only validates physical and bundle links, including broadcast links.
-

- **Prefix Validation**—Prefix validation identifies any forwarding inconsistency of any IGP Prefix SID reachable from the device. The validation is done for all upstream and downstream neighbor combinations of each prefix SID, and identifies inconsistencies in the downstream neighbor. The prefix validation phase simulates customer traffic path by validating both ingress and egress forwarding chain at the DPM processing node.

Since prefix validation is localized to a device running DPM as well as its immediate neighbors, it does not suffer from scale limitations of end-to-end monitoring.

Prefix validation builds on top of adjacency validation by using special MPLS echo requests that travel to the upstream node, return to the DPM-processing node, and time-to-live (TTL) expire at the immediate downstream node, thus exercising entire forwarding path towards the downstream.



Note Observe the following limitations for prefix validation:

- Because prefix validation builds on top of adjacency validation, if a link is not part of adjacency validation, it is not used in prefix validation.
 - If all adjacencies are marked as “Faulty” during adjacency validation, prefix validation is not performed.
 - If a node only has downstream links at a specific node, but no upstream node (possible in certain PE node scenarios), Prefix Validation is not performed.
 - Prefix validation does not support TI-LFA.
-

DPM maintains a database of all prefixes and adjacencies being monitored.

The prefix database is populated by registering as a redistribution client to RIB, which enables DPM to keep the database up-to-date whenever IGP pushes a new prefix SID to RIB, deletes an existing prefix SID, or when the path of an existing prefix SID is modified.

DPM maintains the following prefix data:

- IPv4 Prefix
- Prefix Length
- Prefix SID label

- Error stats

DPM also maintains a list of all local adjacencies. DPM maintains a database that contains local links, their respective local and remote adjacency labels and IP addresses, and error stats.

SR-DPM Operation: Example

The following SR-DPM operation example use the following scenarios:

Figure 23: Test Iteration A Path

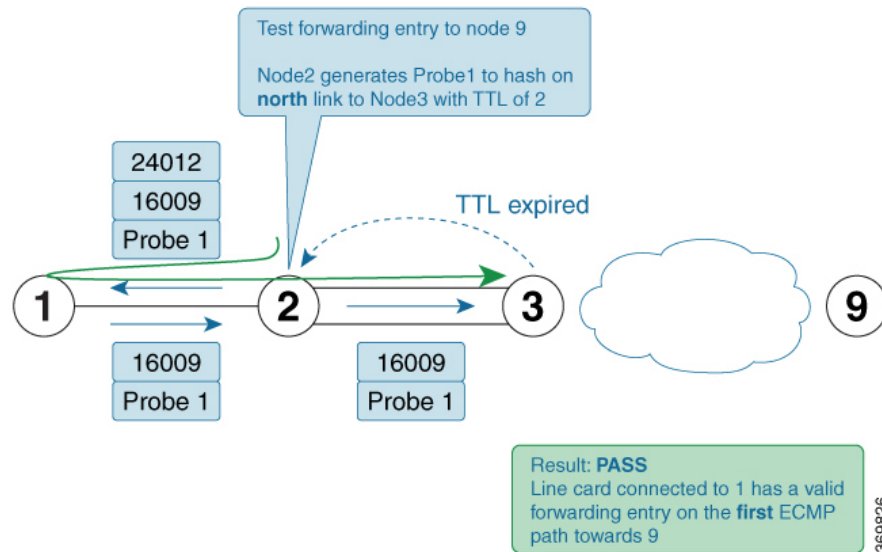
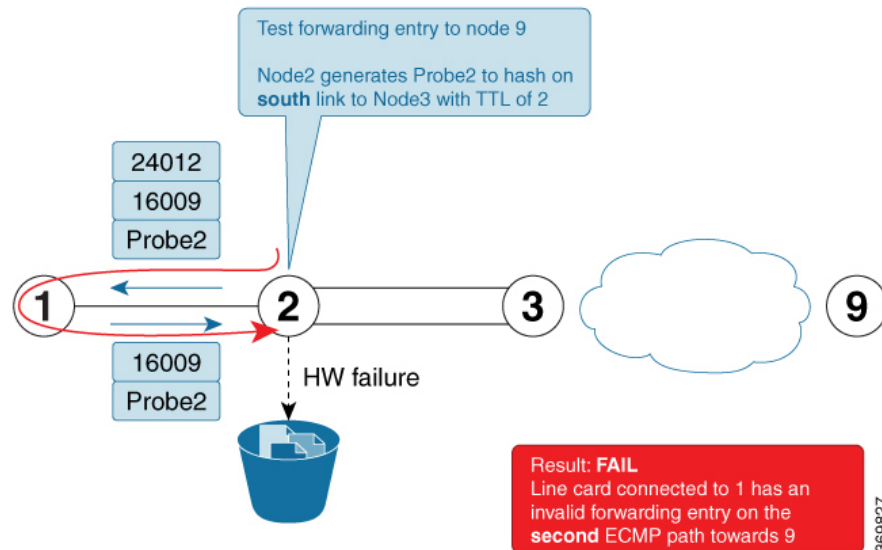


Figure 24: Test Iteration B Path



Node 2 is a DPM-capable device. DPM is enabled *in proactive mode* to perform forwarding consistency tests for all prefix-SIDs in the network. For each destination prefix, the router identifies the directly connected upstream and downstream neighbors used to reach a given destination.

Using node 9 as the prefix under test (prefix-SID = 16009), node 1 is designated as the upstream node and node 3 as the downstream nodes with 2 ECMPs.

1. Node 2 generates test traffic (MPLS OAM ping with source_ip of node 2) to test its forwarding for every upstream/downstream combination. In this case, two combinations exist:
 - Prefix-SID node 9 - test iteration A path = Node 2 to Node 1 to Node 2 to Node 3 (via **top** ECMP)
 - Prefix-SID node 9 - test iteration B path = Node 2 to Node 1 to Node 2 to Node 3 (via **bottom** ECMP)
2. Node 2 adds a label stack in order to enforce the desired path for the test traffic. For example, two labels are added to the packet for test iterations A and B:
 - The top label is equal to the adjacency-SID on node 1 for the interface facing node 2 (adjacency SID = 24012). The bottom label is the prefix-SID under test (16009). The test traffic is sent on the interface facing node 1.
 - The top label (after being POPped at node 1) causes the test traffic to come back to node 2. This returning traffic is completely hardware-switched based on the forwarding entry for the prefix-SID under test (16009). Note that the labeled test traffic has a time-to-live (TTL) of 2 and it will never be forwarded beyond the downstream router(s).
 - When test traffic reaches node 3, a TTL expired response is sent back to node 2. If the response packet arrives over the expected interface (**top** ECMP link) then the forwarding verification on node 2 for the first iteration towards node 9 is considered to be a success.
 - The difference between the test traffic for test iteration A and B in this example is the destination_ip of the MPLS OAM ping. Node 2 calculates them in this order to exercise a given ECMP path (if present). Thus, test traffic for iteration A is hashed onto the **top** ECMP and test traffic for iteration B is hashed onto the **bottom** ECMP link.
3. The DPM tests are then repeated for the remaining prefix-SIDs in the network

Configure SR DPM

To configure SR-DPM, complete the following configurations:

- Enable SR DPM
- Configure SR DPM interval timer
- Configure SR DPM rate limit

Enable SR DPM

Use the **mpls oam dpm** command to enable SR DPM and enter MPLS OAM DPM command mode.

```
Router(config)# mpls oam dpm
Router(config-oam-dpm)#
```

Configure SR DPM Interval Timer

Use the **interval minutes** command in MPLS OAM DPM command mode to specify how often to run DPM scan. The range is from 1 to 3600 minutes. The default is 30 minutes.

```
Router(config-oam-dpm) # interval 240
Router(config-oam-dpm)#
```

Configure SR DPM Rate Limit

Use the `pps pps` command in MPLS OAM DPM command mode to rate limit the number of echo request packets per second (PPS) generated by DPM. The range is from 1 to 250 PPS. The default is 50 PPS.



Note If the specified rate limit is more than the rate limit for overall MPLS OAM requests, DPM generates an error message.

```
Router(config-oam-dpm) # pps 45
Router(config-oam-dpm)#
```

Verification

```
Router# show mpls oam dpm summary
```

Displays the overall status of SR-DPM from the last run.

```
Router# show mpls oam dpm adjacency summary
```

Displays the result of DPM adjacency SID verification for all local interfaces from the last run.

```
Router# show mpls oam dpm adjacency interface
```

Displays the result of DPM adjacency SID verification for all adjacencies for the specified local interface.

```
Router# show mpls oam dpm counters
```

Outputs various counters for DPM from last run as well as since the start of DPM process.

```
Router# show mpls oam dpm prefix summary
```

Displays the result of DPM prefix SID verification for all reachable IGP prefix SIDs from the last run.

```
Router# show mpls oam dpm prefix prefix
```

Displays the result of DPM prefix SID verification for the specified prefix including all upstream and downstream combinations.

```
Router# show mpls oam dpm trace
```

Returns logged traces for DPM.

In addition, the existing `show mpls oam` command is extended to specify DPM counters.

```
Router# show mpls oam counters packet dpm
```