



Segment Routing Configuration Guide for Cisco NCS 5500 Series Routers, IOS XR Release 7.1.x

First Published: 2020-01-29

Last Modified: 2020-08-28

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

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

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

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

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

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

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

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

The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on standards documentation, or language that is used by a referenced third-party product.

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

© 2020 Cisco Systems, Inc. All rights reserved.



CONTENTS

PREFACE

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

CHAPTER 1

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

CHAPTER 2

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

CHAPTER 3

Configure Segment Routing over IPv6 (SRv6)	7
Segment Routing over IPv6 Overview	7
Configuring SRv6 under IS-IS	15
Configuring SRv6 IS-IS TI-LFA	16
SRv6 Services: IPv4 L3VPN	19
SRv6 Services: IPv4 L3VPN Active-Standby Redundancy using Port-Active Mode	23
SRv6 Services for L3VPN Active-Standby Redundancy using Port-Active Mode: Operation	23
Configure SRv6 Services L3VPN Active-Standby Redundancy using Port-Active Mode	24
Configuration Example	24
Running Configuration	24
Verification	25
SRv6 OAM — SID Verification	27

CHAPTER 4	Configure Segment Routing Global Block and Segment Routing Local Block	31
	About the Segment Routing Global Block	31
	About the Segment Routing Local Block	33
	Understanding Segment Routing Label Allocation	34
	Setup a Non-Default Segment Routing Global Block Range	37
	Setup a Non-Default Segment Routing Local Block Range	38

CHAPTER 5	Configure Segment Routing for IS-IS Protocol	41
	Enabling Segment Routing for IS-IS Protocol	41
	Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface	43
	Configuring an Adjacency SID	46
	Manually Configure a Layer 2 Adjacency SID	49
	Configuring Bandwidth-Based Local UCMP	52
	IS-IS Multi-Domain Prefix SID and Domain Stitching: Example	53
	Configure IS-IS Multi-Domain Prefix SID	54
	Configure Common Router ID	54
	Distribute IS-IS Link-State Data	55
	Conditional Prefix Advertisement	55
	Segment Routing ECMP-FEC Optimization	57

CHAPTER 6	Configure Segment Routing for OSPF Protocol	61
	Enabling Segment Routing for OSPF Protocol	61
	Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface	63
	Configuring an Adjacency SID	65
	Segment Routing ECMP-FEC Optimization	67

CHAPTER 7	Configure Segment Routing for BGP	69
	Segment Routing for BGP	69
	Configure BGP Prefix Segment Identifiers	70
	Segment Routing Egress Peer Engineering	71
	Configure Segment Routing Egress Peer Engineering	71
	Configure BGP Link-State	73
	Use Case: Configuring SR-EPE and BGP-LS	77

Configure BGP Proxy Prefix SID 79

CHAPTER 8

Configure SR-TE Policies 83

SR-TE Policy Overview 83

Usage Guidelines and Limitations 84

Instantiation of an SR Policy 84

On-Demand SR Policy – SR On-Demand Next-Hop 84

SR-ODN Configuration Steps 86

Configuring SR-ODN: Examples 88

Configuring SR-ODN for EVPN-VPWS: Use Case 94

Manually Provisioned SR Policy 115

PCE-Initiated SR Policy 115

SR-TE Policy Path Types 115

Dynamic Paths 116

Optimization Objectives 116

Constraints 117

Configure SR Policy with Dynamic Path 119

Anycast SID-Aware Path Computation 120

Explicit Paths 120

SR-TE Policy with Explicit Path 120

Configuring Explicit Path with Affinity Constraint Validation 123

Protocols 125

Path Computation Element Protocol 125

Configure the Head-End Router as PCEP PCC 126

BGP SR-TE 130

Configure BGP SR Policy Address Family at SR-TE Head-End 130

Traffic Steering 132

Automated Steering 132

Color-Only Automated Steering 133

Setting CO Flag 134

Address-Family Agnostic Automated Steering 135

Using Binding Segments 136

Stitching SR-TE Policies Using Binding SID: Example 137

L2VPN Preferred Path 140

Static Route over Segment Routing Policy 141

Miscellaneous 144

LDP over Segment Routing Policy 144

SR-TE MPLS Label Imposition Enhancement 147

SR-TE Reoptimization Timers 149

CHAPTER 9 Segment Routing Tree Segment Identifier 151

Behaviors and Limitations 154

Bud Node Support 154

Configure Segment Routing Tree-SID 155

Running Config 157

CHAPTER 10 Enabling Segment Routing Flexible Algorithm 159

Prerequisites for Flexible Algorithm 159

Building Blocks of Segment Routing Flexible Algorithm 159

Flexible Algorithm Definition 159

Flexible Algorithm Membership 160

Flexible Algorithm Definition Advertisement 160

Flexible Algorithm Prefix-SID Advertisement 160

Calculation of Flexible Algorithm Path 160

Installation of Forwarding Entries for Flexible Algorithm Paths 161

Flexible Algorithm Prefix-SID Redistribution 162

Configuring Flexible Algorithm 162

Example: Configuring IS-IS Flexible Algorithm 163

Example: Configuring OSPF Flexible Algorithm 164

Example: Traffic Steering to Flexible Algorithm Paths 164

BGP Routes on PE – Color Based Steering 164

CHAPTER 11 Configure Segment Routing Path Computation Element 169

About SR-PCE 169

Configure SR-PCE 170

Configure the Disjoint Policy (Optional) 172

Global Maximum-delay Constraint 173

PCE-Initiated SR Policies 174

ACL Support for PCEP Connection	175
Anycast SID-Aware Path Computation	176
PCE Support for MPLS-TE LSPs	179
Configuring the North-Bound API on SR-PCE	182

CHAPTER 12	Configure Performance Measurement	185
	Measurement Modes	186
	Usage Guidelines and Limitations	187
	Link Delay Measurement	188
	SR Policy End-to-End Delay Measurement	199
	SR Policy Liveness Monitoring - Hardware Offloading	206

CHAPTER 13	Configure Topology-Independent Loop-Free Alternate (TI-LFA)	207
	Usage Guidelines and Limitations	209
	Configuring TI-LFA for IS-IS	210
	Configuring TI-LFA for OSPF	212
	TI-LFA Node and SRLG Protection: Examples	214
	Configuring Global Weighted SRLG Protection	215

CHAPTER 14	Configure Segment Routing Microloop Avoidance	219
	About Segment Routing Microloop Avoidance	219
	Usage Guidelines and Limitations	221
	Configure Segment Routing Microloop Avoidance for IS-IS	222
	Configure Segment Routing Microloop Avoidance for OSPF	223

CHAPTER 15	Configure Segment Routing Mapping Server	225
	Segment Routing Mapping Server	225
	Usage Guidelines and Restrictions	226
	Segment Routing and LDP Interoperability	226
	Example: Segment Routing LDP Interoperability	226
	Configuring Mapping Server	229
	Enable Mapping Advertisement	231
	Configure Mapping Advertisement for IS-IS	231
	Configure Mapping Advertisement for OSPF	232

Enable Mapping Client 233

CHAPTER 16**Using Segment Routing OAM 235**

MPLS Ping and Traceroute for BGP and IGP Prefix-SID 235

Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID 236

MPLS LSP Ping and Traceroute Nil FEC Target 238

Examples: LSP Ping and Traceroute for Nil_FEC Target 238

Segment Routing Ping and Traceroute 240

Segment Routing Ping 240

Segment Routing Traceroute 242

Segment Routing Ping and Traceroute for Flexible Algorithm 244

Segment Routing Ping for Flexible Algorithm 244

Segment Routing Traceroute for Flexible Algorithm 245

Segment Routing over IPv6 OAM 245



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 5500 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
January 2020	Initial release of this document
August 2020	Republished for Release 7.1.2

Communications, Services, and Additional Information

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

Cisco Bug Search Tool

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



CHAPTER 1

New and Changed Information for Segment Routing Features

This table summarizes the new and changed feature information for the *Segment Routing Configuration Guide for Cisco NCS 5500 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 7.1.x

Feature	Description	Introduced/Changed in Release	Where Documented
SR-TE IPv4 SR policy over BGPv6 session	This feature is introduced.	Release 7.1.2	BGP SR-TE, on page 130
Segment Routing Performance Measurement - Link Delay Measurement using RFC5357 (TWAMP Light) Encoding	This feature is introduced.	Release 7.1.2	Configure Performance Measurement, on page 185
Segment Routing Performance Measurement - SR Policy End-to-End Delay Measurement	This feature is introduced.	Release 7.1.2	Configure Performance Measurement, on page 185
Segment Routing Ping and Traceroute for Flexible Algorithm	This feature is introduced.	Release 7.1.1	Segment Routing Ping and Traceroute for Flexible Algorithm, on page 244

Feature	Description	Introduced/Changed in Release	Where Documented
Segment Routing Conditional Prefix Advertisement for IS-IS	This feature is introduced.	Release 7.1.1	Conditional Prefix Advertisement, on page 55
Segment Routing Flexible Algorithm Affinity Constraints for IS-IS	This feature is introduced.	Release 7.1.1	Configuring Flexible Algorithm, on page 162
Segment Routing Label Edge Router ECMP-FEC Optimization	This feature is introduced.	Release 7.1.1	Segment Routing ECMP-FEC Optimization, on page 57



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. An adjacency SID can be allocated dynamically from the dynamic label range or configured manually from the segment routing local block (SRLB) range of labels. The adjacency SID is distributed by IS-IS or OSPF. The adjacency segment steers the traffic to a specific adjacency.

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

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

Dataplane

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

Services

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

Segment Routing for Traffic Engineering

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

Need

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

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

Benefits

- **Ready for SDN:** Segment routing was built for SDN and is the foundation for Application Engineered Routing (AER). SR prepares networks for business models, where applications can direct network behavior. SR provides the right balance between distributed intelligence and centralized optimization and programming.
- **Minimal configuration:** Segment routing for TE requires minimal configuration on the source router.
- **Load balancing:** Unlike in RSVP-TE, load balancing for segment routing can take place in the presence of equal cost multiple paths (ECMPs).
- **Supports Fast Reroute (FRR):** Fast reroute enables the activation of a pre-configured backup path within 50 milliseconds of path failure.

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

Workflow for Deploying Segment Routing

Follow this workflow to deploy segment routing.

1. Configure the Segment Routing Global Block (SRGB)
2. Enable Segment Routing and Node SID for the IGP
3. Configure Segment Routing for BGP
4. Configure the SR-TE Policy
5. Configure the SR-PCE
6. Configure TI-LFA and Microloop Avoidance
7. Configure the Segment Routing Mapping Server



CHAPTER 3

Configure Segment Routing over IPv6 (SRv6)

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

- [Segment Routing over IPv6 Overview, on page 7](#)
- [Configuring SRv6 under IS-IS, on page 15](#)
- [Configuring SRv6 IS-IS TI-LFA, on page 16](#)
- [SRv6 Services: IPv4 L3VPN, on page 19](#)
- [SRv6 Services: IPv4 L3VPN Active-Standby Redundancy using Port-Active Mode, on page 23](#)
- [SRv6 OAM — SID Verification, on page 27](#)

Segment Routing over IPv6 Overview

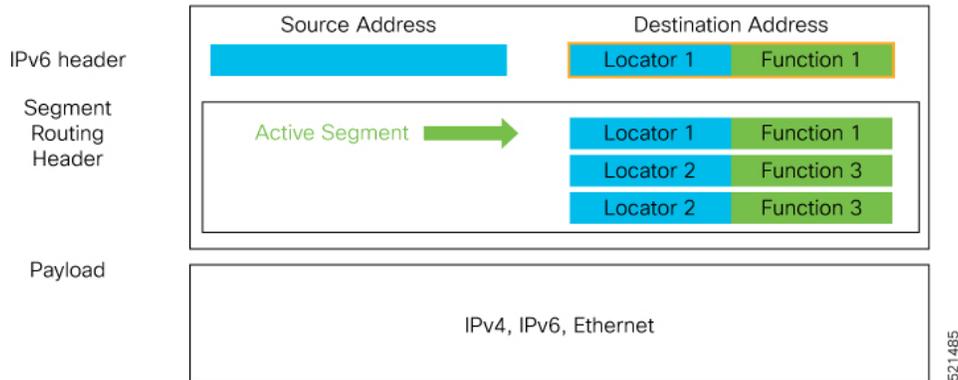
Segment Routing (SR) can be applied on both MPLS and IPv6 data planes. Segment Routing over IPv6 (SRv6) extends Segment Routing support with IPv6 data plane.

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](#).

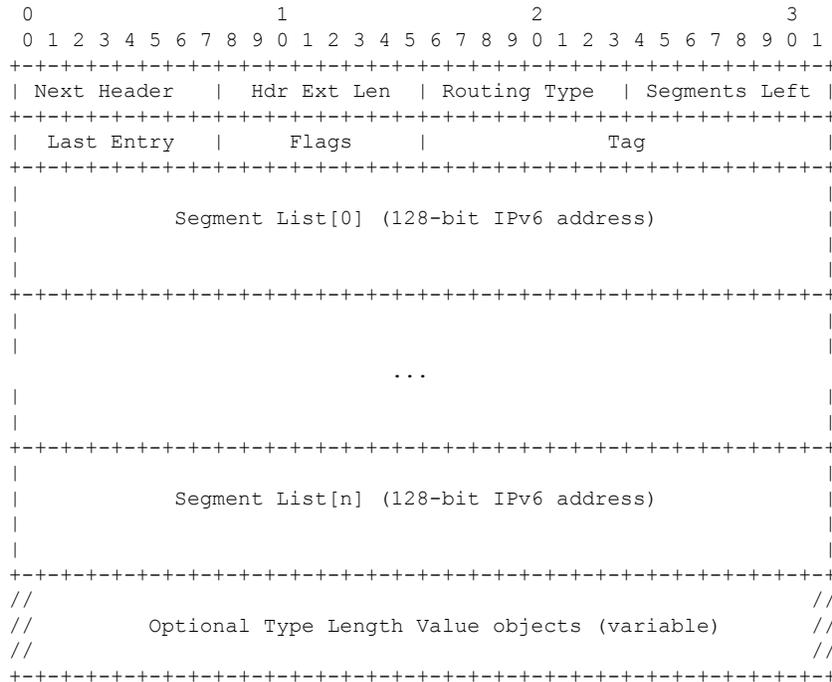
In SRv6, an IPv6 address represents an instruction. SRv6 uses a new type of IPv6 Routing Extension Header, called the Segment Routing Header (SRH), in order to encode an ordered list of instructions. The active segment is indicated by the destination address of the packet, and the next segment is indicated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

Figure 1: Network Program in the Packet Header



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

The SRH is defined as follows:



The following list explains the fields in SRH:

- Next header—Identifies the type of header immediately following the SRH.
- Hdr Ext Len (header extension length)—The length of the SRH in 8-octet units, not including the first 8 octets.
- Segments left—Specifies the number of route segments remaining. That means, the number of explicitly listed intermediate nodes still to be visited before reaching the final destination.
- Last Entry—Contains the index (zero based) of the last element of the segment list.

- Flags— Contains 8 bits of flags.
- Tag—Tag a packet as part of a class or group of packets like packets sharing the same set of properties.
- Segment list—128-bit IPv6 addresses representing the *n*th segment in the segment list. The segment list encoding starts from the last segment of the SR policy (path). That means the first element of the segment list (Segment list [0]) contains the last segment of the SR policy, the second element contains the penultimate segment of the SR policy and so on.

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

- Locator: This is the first part of the SID with most significant bits and represents an address of a specific SRv6 node.
- Function: This is the portion of the SID that is local to the owner node and designates a specific SRv6 function (network instruction) that is executed locally on a particular node, specified by the locator bits.
- Args: This field is optional and represents optional arguments to the function.

The locator part can be further divided into two parts:

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

SRv6 Node Roles

Each node along the SRv6 packet path has a different functionality:

- Source node—A node that can generate an IPv6 packet with an SRH (an SRv6 packet), or an ingress node that can impose an SRH on an IPv6 packet.
- Transit node—A node along the path of the SRv6 packet (IPv6 packet and SRH). The transit node does not inspect the SRH. The destination address of the IPv6 packet does not correspond to the transit node.
- Endpoint node—A node in the SRv6 domain where the SRv6 segment is terminated. The destination address of the IPv6 packet with an SRH corresponds to the end point node. The segment endpoint node executes the function bound to the SID

SRv6 Head-End Behaviors

The SR Headend with Encapsulation behaviors are documented in the [IETF RFC 8986 SRv6 Network Programming](#).

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

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

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

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

- H.Insert.Red—H.Insert with reduced insertion

SRv6 Endpoint Behaviors

The SRv6 endpoint behaviors are documented in the [IETF RFC 8986 SRv6 Network Programming](#).

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

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

SRv6 Endpoint Behavior Variants

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.

Usage Guidelines and Limitations

General Guidelines and Limitations

- Cisco IOS XR supports the following SRv6 SID behaviors and variants:
 - END with PSP
 - END.X with PSP
 - END.DT4
 - END.DT6
- SRv6 Underlay support includes:

- IGP redistribution/leaking between levels
- Prefix Summarization on ABR routers
- IS-IS TI-LFA
- Microloop Avoidance
- Flex-algo

Configuring SRv6

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

The following usage guidelines and restrictions apply while configuring SRv6.

- All routers in the SRv6 domain should have the same SID block (network designator) in their locator.
- The locator length should be 64-bits long.
 - The SID block portion (MSBs) cannot exceed 40 bits. If this value is less than 40 bits, user should use a pattern of zeros as a filler.
 - The Node Id portion (LSBs) cannot exceed 24 bits.
- You can configure up to 8 locators to support SRv6 Flexible Algorithm. All locators prefix must share the same SID block (first 40-bits).

Enabling SRv6 with Locator

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

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

Optional: Configuring Encapsulation Parameters

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

- Source Address of outer encapsulating IPv6 header: The default source address for encapsulation is one of the loopback addresses.
- The hop limit of outer-encapsulating IPv6 header. The range is from 1 to 254 for NCS 5500 and from 1 to 255 for NCS 5700; the default value for hop-limit is 254.

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

Optional: Enabling Syslog Logging for Locator Status Changes

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

```
Router(config)# segment-routing srv6
Router(config-srv6)# logging locator status
```

Verifying SRv6 Manager

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

```
Router# show segment-routing srv6 manager
Parameters:
  Parameters:
    SRv6 Enabled: Yes
    SRv6 Operational Mode:
      Base:
        SID Base Block: 2001:db8::/40
  Encapsulation:
    Source Address:
      Configured: 1::1
      Default: 5::5
    Hop-Limit: Default
    Traffic-class: Default
Summary:
  Number of Locators: 1 (1 operational)
  Number of SIDs: 4 (0 stale)
  Max SIDs: 64000
  OOR:
    Thresholds: Green 3200, Warning 1920
    Status: Resource Available
      History: (0 cleared, 0 warnings, 0 full)
    Block 2001:db8:0:a2::/64:
      Number of SIDs free: 65470
      Max SIDs: 65470
      Thresholds: Green 3274, Warning 1965
      Status: Resource Available
      History: (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: Yes
  Microloop-Avoidance: Yes
  Endpoint behaviors:
    End (PSP)
    End.X (PSP)
    End.DX6
    End.DX4
    End.DT6
    End.DT4
    End.DX2
    uN (PSP/USD)
    uA (PSP/USD)
    uDT6
    uDT4
    uDX2
    uB6 (Insert.Red)
  Headend behaviors:
    T
    H.Insert.Red
    H.Encaps.Red
  Security rules:
    SEC-1
    SEC-2
    SEC-3
  Counters:
    CNT-1
    CNT-3
  Signaled parameters:
    Max-SL : 3
```

```

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

```

Verifying SRv6 Locator

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

```

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

```

Verifying SRv6 Local SIDs

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

```

Router# show segment-routing srv6 locator myLoc1 sid
SID           State  RW      Function      Context           Owner
-----
2001:db8:0:a2:1::
  InUse      Y
2001:db8:0:a2:40::
  InUse      Y
2001:db8:0:a2:41::
  InUse      Y

```

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

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

```

Router# show segment-routing srv6 locator myLoc1 sid 2001:db8:0:a2:40:: detail
SID           State  RW      Function      Context           Owner
-----
2001:db8:0:a2:40::
  InUse      Y
  SID context: { table-id=0xe0000011 ('VRF1':IPv4/Unicast) }
  Locator: myLoc1'
  Allocation type: Dynamic

```

Created: Feb 1 14:04:02.901 (3d00h ago)

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

show Commands

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

Command	Description
show segment-routing srv6 manager	Displays the summary information from SRv6 manager, including platform capabilities.
show segment-routing srv6 locator <i>locator-name</i> [detail]	Displays the SRv6 locator information on the router.
show segment-routing srv6 locator <i>locator-name</i> sid [[<i>sid-ipv6-address</i> [detail]	Displays the information regarding SRv6 local SID(s) allocated from a given locator.
show segment-routing srv6 sid [<i>sid-ipv6-address</i> all stale] [detail]	Displays SID information across locators. By default, only “active” (i.e. non-stale) SIDs are displayed.
show route ipv6 local-srv6	Displays all SRv6 local-SID prefixes in IPv6 RIB.

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 5500*.

Configuring SRv6 IS-IS TI-LFA

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

TI-LFA provides link protection in topologies where other fast reroute techniques cannot provide protection. The goal of TI-LFA is to reduce the packet loss that results while routers converge after a topology change due to a link failure. TI-LFA leverages the post-convergence path which is planned to carry the traffic and ensures link and node protection within 50 milliseconds. TI-LFA with IS-IS SR-MPLS is already supported.

Usage Guidelines and Restrictions

The following usage guidelines and restrictions apply:

Configuring SRv6 IS-IS TI-LFA

The following example shows how to configure SRv6 IS-IS TI-LFA.



Note Complete the [Configuring SRv6](#) before performing these steps.

```
Router(config)# router isis core
Router(config-isis)# address-family ipv6 unicast
Router(config-isis-af)# segment-routing srv6
Router(config-isis-srv6)# locator locator1
Router(config-isis-srv6-loc)# exit
Router(config-isis)# interface loopback 0
Router(config-isis-if)# passive
Router(config-isis-if)# address-family ipv6 unicast
Router(config-isis-if-af)# exit
Router(config-isis)# interface bundle-ether 1201
Router(config-isis-if)# address-family ipv6 unicast
Router(config-isis-if-af)# fast-reroute per-prefix
Router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
Router(config-isis-if-af)# exit
Router(config-isis)# interface bundle-ether 1301
Router(config-isis-if)# address-family ipv6 unicast
```

```
Router(config-isis-if-af)# fast-reroute per-prefix
Router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
Router(config-isis-if-af)# exit
```

Verification

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

```
Router# show isis ipv6 fast-reroute cafe:0:0:66::/64 detail
Thu Nov 22 16:12:51.983 EST

L1 cafe:0:0:66::/64 [11/115] low priority
  via fe80::2, TenGigE0/0/0/6, SRv6-HUB6, Weight: 0
  Backup path: TI-LFA (link), via fe80::1, Bundle-Ether1201 SRv6-LF1, Weight: 0, Metric:
  51
    P node: SRv6-TP8.00 [8::8], SRv6 SID: cafe:0:0:88:1:: End (PSP)
    Backup-src: SRv6-HUB6.00
    P: No, TM: 51, LC: No, NP: No, D: No, SRLG: Yes
    src SRv6-HUB6.00-00, 6::6
```

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

```
Router# show route ipv6 cafe:0:0:66::/64 detail
Thu Nov 22 16:14:07.385 EST

Routing entry for cafe:0:0:66::/64
  Known via "isis srv6", distance 115, metric 11, type level-1
  Installed Nov 22 09:24:05.160 for 06:50:02
  Routing Descriptor Blocks
    fe80::2, from 6::6, via TenGigE0/0/0/6, Protected
      Route metric is 11
      Label: None
      Tunnel ID: None
      Binding Label: None
      Extended communities count: 0
      Path id:1          Path ref count:0
      NHID:0x2000a(Ref:11)
      NHID eid:0xfffffffffffffffff
      SRv6 Headend: H.Insert.Red [base], SRv6 SID-list {cafe:0:0:88:1::}
      Backup path id:65
    fe80::1, from 6::6, via Bundle-Ether1201, Backup (TI-LFA)
      Repair Node(s): 8::8
      Route metric is 51
      Label: None
      Tunnel ID: None
      Binding Label: None
      Extended communities count: 0
      Path id:65          Path ref count:1
      NHID:0x2000d(Ref:11)
      NHID eid:0xfffffffffffffffff
      SRv6 Headend: H.Insert.Red [base], SRv6 SID-list {cafe:0:0:88:1::}
      MPLS eid:0x1380800000001
```

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

```

Router# show cef ipv6 cafe:0:0:66::/64 detail location 0/0/cpu0
Thu Nov 22 17:01:58.536 EST
cafe:0:0:66::/64, version 1356, SRv6 Transit, internal 0x1000001 0x2 (ptr 0x8a4a45cc) [1],
 0x0 (0x8a46ae20), 0x0 (0x8c8f31b0)
Updated Nov 22 09:24:05.166
local adjacency fe80::2
Prefix Len 64, traffic index 0, precedence n/a, priority 2
  gateway array (0x8a2dfaf0) reference count 4, flags 0x500000, source rib (7), 0 backups
    [5 type 3 flags 0x8401 (0x8a395d58) ext 0x0 (0x0)]
  LW-LDI[type=3, refc=1, ptr=0x8a46ae20, sh-ldi=0x8a395d58]
  gateway array update type-time 1 Nov 22 09:24:05.163
  LDI Update time Nov 22 09:24:05.163
  LW-LDI-TS Nov 22 09:24:05.166
    via fe80::2/128, TenGigE0/0/0/6, 8 dependencies, weight 0, class 0, protected [flags
0x400]
      path-idx 0 bkup-idx 1 NHID 0x2000a [0x8a2c2fd0 0x0]
      next hop fe80::2/128
    via fe80::1/128, Bundle-Ether1201, 8 dependencies, weight 0, class 0, backup (TI-LFA)
[flags 0xb00]
      path-idx 1 NHID 0x2000d [0x8c2670b0 0x0]
      next hop fe80::1/128, Repair Node(s): 8::8
      local adjacency
        SRv6 H.Insert.Red SID-list {cafe:0:0:88:1::}

Load distribution: 0 (refcount 5)

Hash OK Interface Address
0 Y TenGigE0/0/0/6 fe80::2

```

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

```

Router# show cef ipv6 fast-reroute-db
Sun Dec 9 20:23:08.111 EST

PROTECT-FRR: per-prefix [1, 0x0, 0x0, 0x98c83270]
protect-interface: Te0/0/0/6 (0x208)
protect-next-hop: fe80::2/128
ipv6 nhinfo [0x977397d0]
Update Time Dec 9 17:29:42.427

  BACKUP-FRR: per-prefix [5, 0x0, 0x2, 0x98c83350]
  backup-interface: BE1201 (0x800002c)
  backup-next-hop: fe80::1/128
  ipv6 nhinfo [0x977396a0 protect-frr: 0x98c83270]
Update Time Dec 9 17:29:42.428

PROTECT-FRR: per-prefix [1, 0x0, 0x0, 0x98c830b0]
protect-interface: BE1201 (0x800002c)
protect-next-hop: fe80::1/128
ipv6 nhinfo [0x977396a0]
Update Time Dec 9 17:29:42.429

  BACKUP-FRR: per-prefix [5, 0x0, 0x1, 0x98c83190]
  backup-interface: Te0/0/0/6 (0x208)
  backup-next-hop: fe80::2/128
  ipv6 nhinfo [0x977397d0 protect-frr: 0x98c830b0]
Update Time Dec 9 17:29:42.429

```

SRv6 Services: IPv4 L3VPN

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

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

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

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

Usage Guidelines and Limitations

- SRv6 locator can be assigned globally, for all VRFs, or for an individual VRF.
- Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
- BGP, OSPF, Static are supported as PE-CE protocol.
- MPLS L3VPN and SRv6 L3VPN interworking gateway is not supported.

Configuring SRv6 based IPv4 L3VPN

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

Configure an Individual VRF with Per-VRF Label Allocation Mode

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

Configure an Individual VRF with Per-CE Label Allocation Mode

```
RP/0/0/CPU0:Router(config-bgp-af)# vrf vrf2
RP/0/0/CPU0:Router(config-bgp-vrf)# rd 106:2
RP/0/0/CPU0:Router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-bgp-vrf-af)# segment-routing srv6
RP/0/0/CPU0:Router(config-bgp-vrf-af-srv6)# alloc mode per-ce
```

```
RP/0/0/CPU0:Router(config-bgp-vrf-af-srv6)# exit
RP/0/0/CPU0:Router(config-bgp-vrf-af)# exit
RP/0/0/CPU0:Router(config-bgp-vrf)# neighbor 10.1.2.2
RP/0/0/CPU0:Router(config-bgp-vrf-nbr)# remote-as 100
RP/0/0/CPU0:Router(config-bgp-vrf-nbr)# address-family ipv4 unicast
```

Verification

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

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

```
RP/0/0/CPU0:SRv6-Hub6# show segment-routing srv6 sid
*** Locator: 'my_locator' ***
SID          State  RW          Function      Context          Owner
-----
-----
cafe:0:0:66:1::      InUse  Y          End (PSP)     'my_locator':1  sidmgr
cafe:0:0:66:40::    InUse  Y          End.X (PSP)   [Te0/0/0/2, Link-Local]  isis-srv6
cafe:0:0:66:41::    InUse  Y          End.X (PSP)   [BE6801, Link-Local]    isis-srv6
cafe:0:0:66:42::    InUse  Y          End.X (PSP)   [BE5601, Link-Local]    isis-srv6
cafe:0:0:66:43::    InUse  Y          End.X (PSP)   [BE5602, Link-Local]    isis-srv6
cafe:0:0:66:44::    InUse  Y          End.DT4       'VRF1'          bgp-100
cafe:0:0:66:45::    InUse  Y          End.DT4       'VRF2'          bgp-100
cafe:0:0:66:46::    InUse  Y          End.DX4       'VRF2':3        bgp-100
cafe:0:0:66:47::    InUse  Y          End.DX4       'VRF2':4        bgp-100
```

The following example shows how to verify the SRv6 based L3VPN configuration using the **show segment-routing srv6SID-prefixdetail** command.

```
RP/0/RP0/CPU0:SRv6-Hub6# show segment-routing srv6 sid cafe:0:0:66:44:: detail
Sun Dec  9 16:52:54.015 EST
*** Locator: 'my_locator' ***
SID          State  RW          Function      Context          Owner
-----
-----
cafe:0:0:66:44::    InUse  Y          End.DT4       'VRF1'          bgp-100
SID context: { table-id=0xe0000001 ('VRF1':IPv4/Unicast) }
Locator: 'my_locator'
Allocation type: Dynamic
Created: Dec  8 16:34:32.506 (1d00h ago)

RP/0/RP0/CPU0:SRv6-Hub6# show segment-routing srv6 sid cafe:0:0:66:47:: detail
Sun Dec  9 16:54:26.073 EST
*** Locator: 'my_locator' ***
SID          Function      Context          Owner
```


The following example shows how to verify the SRv6 based L3VPN configuration using the **show route vrfvrf-name/prefixdetail** command.

```
RP/0/RP0/CPU0:SRv6-LF1# show route vrf VRF1 10.15.0.0/30 detail
Wed Nov 21 16:35:17.775 EST
Routing entry for 10.15.0.0/30
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Installed Nov 21 16:35:14.107 for 00:00:03
Routing Descriptor Blocks
  6::6, from 2::2
    Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
    Route metric is 0
    Label: None
    Tunnel ID: None
    Binding Label: None
    Extended communities count: 0
    Source RD attributes: 0x0000:106:1
    NHID:0x0(Ref:0)
    SRv6 Headend: H.Encaps.Red [base], SID-list { cafe:0:0:66:44:: }
    MPLS eid:0x1380600000001
  Route version is 0xd (13)
  No local label
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type RIB_SVD_TYPE_REMOTE
  Download Priority 3, Download Version 3038384
  No advertising protos.
```

The following example shows how to verify the SRv6 based L3VPN configuration for per-ce allocation mode using the **show bgp vrfvrf-namexthop-set** command.

```
RP/0/RP0/CPU0:SRv6-Hub6# show bgp vrf VRF2 nexthop-set
Wed Nov 21 15:52:17.464 EST
Resilient per-CE nexthop set, ID 3
Number of nexthops 1, Label 0, Flags 0x2200
SRv6-VPN SID: cafe:0:0:66:46::/128
Nexthops:
10.1.2.2
Reference count 1,
Resilient per-CE nexthop set, ID 4
Number of nexthops 2, Label 0, Flags 0x2100
SRv6-VPN SID: cafe:0:0:66:47::/128
Nexthops:
10.1.2.2
10.2.2.2
Reference count 2,
```

The following example shows how to verify the SRv6 based L3VPN configuration using the **show cef vrfvrf-name prefix detail locationline-card** command.

```
RP/0/RP0/CPU0:SRv6-LF1# show cef vrf VRF1 10.15.0.0/30 detail location 0/0/cpu0
Wed Nov 21 16:37:06.894 EST
151.1.0.0/30, version 3038384, SRv6 Transit, internal 0x5000001 0x0 (ptr 0x9ae6474c) [1],
0x0 (0x0), 0x0 (0x8c11b238)
Updated Nov 21 16:35:14.109
Prefix Len 30, traffic index 0, precedence n/a, priority 3
gateway array (0x8cd85190) reference count 1014, flags 0x2010, source rib (7), 0 backups
[1 type 3 flags 0x40441 (0x8a529798) ext 0x0 (0x0)]
```

```

LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
gateway array update type-time 1 Nov 21 14:47:26.816
LDI Update time Nov 21 14:52:53.073
Level 1 - Load distribution: 0
[0] via cafe:0:0:66::/128, recursive
via cafe:0:0:66::/128, 7 dependencies, recursive [flags 0x6000]
path-idx 0 NHID 0x0 [0x8acb53cc 0x0]
next hop VRF - 'default', table - 0xe0800000
next hop cafe:0:0:66::/128 via cafe:0:0:66::/64
  SRv6 H.Encaps.Red SID-list {cafe:0:0:66:44::}
Load distribution: 0 (refcount 1)
Hash OK Interface Address
0 Y Bundle-Ether1201 fe80::2

```

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

The Segment Routing IPv6 (SRv6) Services: IPv4 L3VPN Active-Standby Redundancy using Port-Active Mode feature provides all-active per-port load balancing for multihoming. The forwarding of traffic is determined based on a specific interface rather than per-flow across multiple Provider Edge routers. This feature enables efficient load-balancing and provides faster convergence. In an active-standby scenario, the active PE router is detected using designated forwarder (DF) election by modulo calculation and the interface of the standby PE router brought down. For Modulo calculation, byte 10 of the Ethernet Segment Identifier (ESI) is used.

Usage Guidelines and Restrictions

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

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

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

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

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

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

Configuration Example

```

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

/* Configure load balancing. */
Router# configure
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether10
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 11.11.11.11.11.11.11.14
Router(config-evpn-ac-es)# load-balancing-mode port-active
Router(config-evpn-ac-es)# commit
!
/* Configure address family session in BGP. */
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 192.168.0.2
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp)# neighbor 192.168.0.3
Router(config-bgp-nbr)# remote-as 200
Router(config-bgp-nbr)# update-source Loopback 0
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr)# commit

```

Running Configuration

```

interface Bundle-Ether14
  ipv4 address 14.0.0.2 255.255.255.0
  ipv6 address 14::2/64
  lACP period short
  mac-address 1.2.3
  bundle wait-while 0
!
interface GigabitEthernet0/2/0/5
  bundle id 14 mode active
!
evpn
  interface Bundle-Ether14
    ethernet-segment
      identifier type 0 11.11.11.11.11.11.11.14
      load-balancing-mode port-active

```

```

!
!
!
router bgp 100
  bgp router-id 192.168.0.2
  address-family l2vpn evpn
  !
  neighbor 192.168.0.3
    remote-as 100
    update-source Loopback0
    address-family l2vpn evpn
  !
!
!
!

```

Verification

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

```

/* Verify ethernet-segment details on active DF router */
Router# show evpn ethernet-segment interface Bundle-Ether14 detail
Ethernet Segment Id      Interface      Nexthops
-----
0011.1111.1111.1111.1114 BE14          192.168.0.2
                                192.168.0.3

    ES to BGP Gates      : Ready
    ES to L2FIB Gates   : Ready
    Main port           :
      Interface name    : Bundle-Ether14
      Interface MAC     : 0001.0002.0003
      IfHandle          : 0x000041d0
      State              : Up
      Redundancy        : Not Defined
    ESI type            : 0
      Value              : 11.1111.1111.1111.1114
    ES Import RT        : 1111.1111.1111 (from ESI)
    Source MAC          : 0000.0000.0000 (N/A)
    Topology            :
      Operational       : MH
      Configured        : Port-Active
    Service Carving     : Auto-selection
      Multicast         : Disabled
    Peering Details     :
      192.168.0.2 [MOD:P:00]
      192.168.0.3 [MOD:P:00]

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

/* Verify bundle Ethernet configuration on active DF router */
Router# show bundle bundle-ether 14
Bundle-Ether14
  Status:                               Up

```

```

Local links <active/standby/configured>: 1 / 0 / 1
Local bandwidth <effective/available>: 1000000 (1000000) kbps
MAC address (source): 0001.0002.0003 (Configured)
Inter-chassis link: No
Minimum active links / bandwidth: 1 / 1 kbps
Maximum active links: 64
Wait while timer: Off
Load balancing:
  Link order signaling: Not configured
  Hash type: Default
  Locality threshold: None
LACP: Operational
  Flap suppression timer: Off
  Cisco extensions: Disabled
  Non-revertive: Disabled
mLACP: Not configured
IPv4 BFD: Not configured
IPv6 BFD: Not configured

```

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

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

```
Router# show evpn ethernet-segment interface bundle-ether 10 detail
```

```

Router# show evpn ethernet-segment interface Bundle-Ether24 detail
Ethernet Segment Id      Interface      Nexthops
-----
0011.1111.1111.1111.1114 BE24          192.168.0.2
                                192.168.0.3

```

```

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

```

```

Service Carving Results:
  Forwarders : 0
  Permanent : 0
  Elected : 0
  Not Elected : 0
MAC Flushing mode : STP-TCN
Peering timer : 3 sec [not running]
Recovery timer : 30 sec [not running]
Carving timer : 0 sec [not running]

```

```

Local SHG label   : None
Remote SHG labels : 0

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

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

Port          Device          State          Port ID          B/W, kbps
-----
Gi0/0/0/4    Local          Standby       0x8000, 0x0002  1000000
Link is in standby due to bundle out of service state

```

SRv6 OAM — SID Verification

Use the following commands to perform SRv6 ping and traceroute:

```
ping B:k:F:: [use-srv6-op-sid [ end.op-sid-value]]
```

```
traceroute B:k:F:: [use-srv6-op-sid [ end.op-sid-value]]
```

Where *B:k:F::* is the target SID at node *k* with locator block *B* and function *F*.

Ping/Traceroute to SID Without OAM SID

The user can issue ping or traceroute to an SRv6 SID using the classic CLI. A ping or traceroute to an SRv6 SID does not require the user to enter the “use-end-op” keyword when pinging or tracing a SID function. In this case, and as usual, the packet is pre-routed as an ICMP echo request or UDP packet.

Ping/Traceroute to SID With OAM SID

When ping or traceroute operations include the **use-srv6-op-sid** keyword, the packet is pre-routed with END.OP SID as Destination Address (DA) and the target SID in the SRH.



Note The END.OP SID value is an optional 128 bit value in IPv6 address format. See **END.OP SID Derivation** below for details on this value.

At the target node, the END.OP SID forces the punt of the packet to the OAM process, which verifies that the next SID is local and valid. If the next SID received by the target node is a local valid address that is not a SID, the target node still replies to indicate ping success. The ping reply contains a subtype to indicate the target was a SID or a local address.

A target remote SID include the following:

- END
- END.DT4/END.DX4 (used by L3 Services over SRv6)

END.OP SID Derivation

The ingress node can automatically derive the END.OP SID associated with a specified target SID by leveraging the IGP topology database in that node. The database will contain END.OP SIDs from remote nodes.

An END.OP SID associated with a locator will be advertised by IS-IS within an IGP domain in an area/level, which is added to the topology database. However, END.OP SIDs are not redistributed by IS-IS across IGP domains or across different area/level within an IGP domain. In this case, the topology database in a node contains END.OP SIDs only from the nodes within the same IGP domain in an area/level. An END.OP SID cannot be determine automatically if the specified target SID is external to the domain. For target SIDs across IGP domains or across different area/level within an IGP domain, the *end.op-sid-value* must be explicitly provided.

If *end.op-sid-value* is not provided and the END.OP SID cannot be automatically derived, an error is displayed prompting the user to provide the *end.op-sid-value*.

Configuration Examples

The following example shows using ping to a SID without OAM SID.

```
Router# ping cafe:0:0:a3:40::
Wed Jul 24 19:24:50.812 UTC
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to cafe:0:0:a3:40::, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/2 ms
```

The following example shows using ping to a SID with an OAM SID. Note that the output shows "S" to indicate that this is a response from a SID target.

```
Router# ping cafe:0:0:a3:40:: use-srv6-op-sid
Wed Jul 24 19:24:50.812 UTC
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to cafe:0:0:a3:40::, timeout is 2 seconds:
SSSSS
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/2 ms
```

The following example shows using ping to a SID with an explicit OAM SID. Note that the output shows "S" to indicate that this is a response from a SID target.

```
Router# ping cafe:0:0:a3:40:: use-srv6-op-sid cafe:0:0:a3:11::
Wed Jul 24 19:24:50.812 UTC
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to cafe:0:0:a3:40::, timeout is 2 seconds:
SSSSS
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/2 ms
```

The following example shows using traceroute to a SID without OAM SID.

```

Router# traceroute cafe:0:0:a3:1::
Wed Jul 24 19:40:19.192 UTC

Type escape sequence to abort.
Tracing the route to cafe:0:0:a3:1::

 1  2001::1:1:1:1 2 msec 2 msec 2 msec
 2  2001:10:10:13::2 2 msec 2 msec 2 msec

```

The following example shows using traceroute to a SID with OAM SID.

```

Router# traceroute cafe:0:0:a3:40:: use-srv6-op-sid

Type escape sequence to abort.
Tracing the route to cafe:0:0:a3:40::

 1  2001::1:1:1:1
    [IP tunnel: DA=cafe:0:0:a3:11:: SRH Stack 0 =(cafe:0:0:a3:40:: ,SL=1) ] 2
msec
 2  2001::33:33:33:33
    [IP tunnel: DA=cafe:0:0:a3:11:: SRH Stack 0 =(cafe:0:0:a3:40:: ,SL=1) ] 3
msec

```

The following example shows using traceroute to a SID with an explicit OAM SID.

```

Router# traceroute cafe:0:0:a3:40:: use-srv6-op-sid cafe:0:0:a3:11::

Type escape sequence to abort.
Tracing the route to cafe:0:0:a3:40::

 1  2001::1:1:1:1
    [IP tunnel: DA=cafe:0:0:a3:11:: SRH Stack 0 =(cafe:0:0:a3:40:: ,SL=1) ] 2
msec
 2  2001::33:33:33:33
    [IP tunnel: DA=cafe:0:0:a3:11:: SRH Stack 0 =(cafe:0:0:a3:40:: ,SL=1) ] 3
msec

```




CHAPTER 4

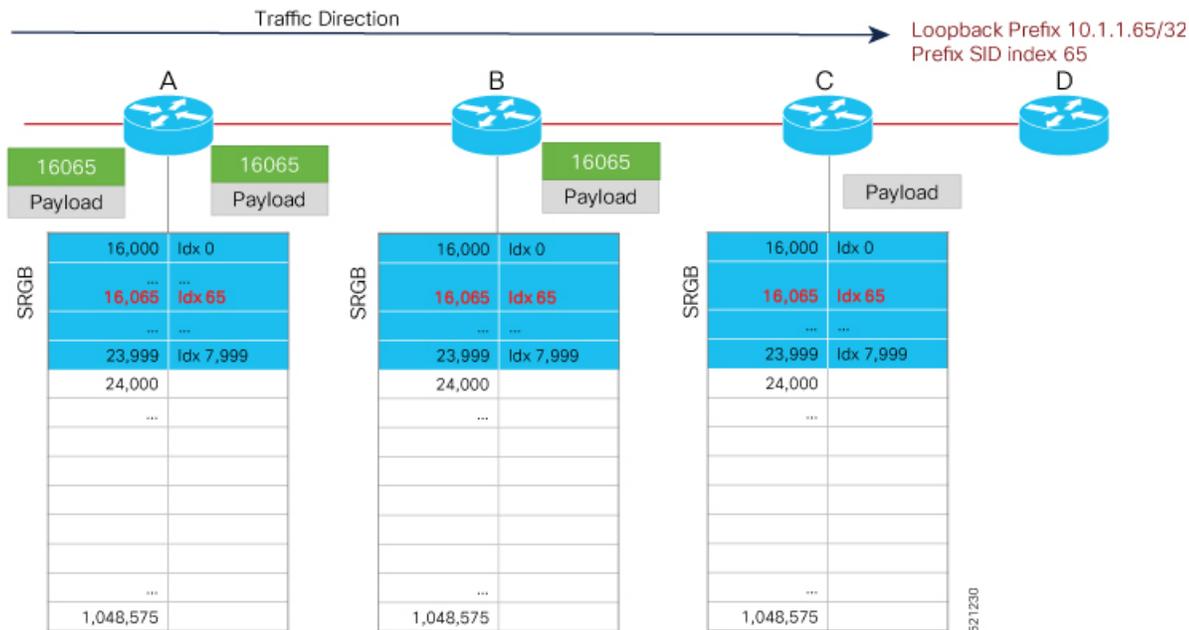
Configure Segment Routing Global Block and Segment Routing Local Block

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

- [About the Segment Routing Global Block, on page 31](#)
- [About the Segment Routing Local Block, on page 33](#)
- [Understanding Segment Routing Label Allocation, on page 34](#)
- [Setup a Non-Default Segment Routing Global Block Range, on page 37](#)
- [Setup a Non-Default Segment Routing Local Block Range, on page 38](#)

About the Segment Routing Global Block

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



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

Behaviors and Limitations

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

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

SRGB Label Conflicts

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

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

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

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

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



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



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



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

About the Segment Routing Local Block

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

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

Behaviors and Limitations

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

SRLB Label Conflicts

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

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



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



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



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

Understanding Segment Routing Label Allocation

In IOS XR, local label allocation is managed by the Label Switching Database (LSD). MPLS applications must register as a client with the LSD to allocate labels. Most MPLS applications (for example: LDP, RSVP, L2VPN, BGP [LU, VPN], IS-IS and OSPF [Adj-SID], SR-TE [Binding-SID]) use labels allocated dynamically by LSD.

With Segment Routing-capable IOS XR software releases, the LSD *preserves* the default SRLB label range (15,000 to 15,999) and default SRGB label range (16,000 to 23,999), even if Segment Routing is not enabled.

This preservation of the default SRLB/SRGB label range makes future Segment Routing activation possible without a reboot. No labels are allocated from this preserved range. When you enable Segment Routing with the default SRLB/SRGB in the future, these label ranges will be available and ready for use.

The LSD allocates dynamic labels starting from 24,000.



Note If an MPLS label range is configured and it overlaps with the default SRLB/SRGB label ranges (for example, **mpls label range 15000 1048575**), then the default SRLB/SRGB preservation is disabled.

Example 1: LSD Label Allocation When SR is not Configured

- Special use: 0-15
- MPLS static: 16 to 14,999
- SRLB (preserved): 15,000 to 15,999
- SRGB (preserved): 16,000 to 23,999
- Dynamic: 24,000 to max



Example 2: LSD Label Allocation When SR is Configured with Default SRGB and Default SRLB

- Special use: 0-15
- MPLS static: 16 to 14,999
- SRLB (reserved): 15,000 to 15,999
- SRGB (reserved): 16,000 to 23,999
- Dynamic: 24,000 to max

0	Special-purpose and MPLS static labels
...	
14,999	
15,000 - 15,999	Reserved range (SRLB)
16,000	Reserved range (SRGB)
...	
23,999	
24,000	Dynamic label range
...	
...	
...	
...	
...	
...	
...	
...	
1,048,575	

521228

Example 3: LSD Label Allocation When SR is Configured with Non-default SRGB and Non-default SRLB

- Special use: 0-15
- MPLS static: 16 to 14,999
- SRLB (preserved): 15,000 to 15,999
- SRGB (preserved): 16,000 to 23,999
- Dynamic: 24000 to 28,999
- SRLB (reserved): 29,000 to 29,999
- SRGB (reserved): 30,000 to 39,999
- Dynamic: 40,000 to max

0	
...	Special-purpose and MPLS static labels
14,999	
15,000 - 15,999	Preserved range (SRLB)
16,000	
...	Preserved range (SRGB)
23,999	
24,000	
...	Dynamic label range
28,999	
29,000 - 29,999	Reserved range (SRLB)
30,000	
...	Reserved range (SRGB)
39,999	
40,000	
...	Dynamic label range
...	
1,048,575	521,229

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 global configuration 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:

	Command or Action	Purpose
		<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.

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

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

What to do next

Configure prefix SIDs and enable segment routing.

Setup a Non-Default Segment Routing Local Block Range

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

SUMMARY STEPS

1. **configure**
2. **segment-routing local-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# configure	Enters global configuration mode.
Step 2	segment-routing local-block <i>starting_value ending_value</i> Example: RP/0/RP0/CPU0:router(config)# segment-routing local-block 30000 30999	Enter the lowest value that you want the SRLB range to include as the starting value. Enter the highest value that you want the SRLB range to include as the ending value.

	Command or Action	Purpose
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.

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

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

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

Router# show segment-routing local-block inconsistencies

No inconsistencies
```

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

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

%ROUTING-MPLS_LSD-3-ERR_SRLB_RANGE : SRLB allocation failed: 'SRLB reservation not successful
for [30000,30999]. Use with caution 'clear segment-routing local-block discrepancy all'
command
to force srlb allocation'
```



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

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

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

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

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

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

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

```
Router# reload

Proceed with reload? [confirm]yes
```

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

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

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

Router# show segment-routing local-block inconsistencies

No inconsistencies
```

What to do next

Configure adjacency SIDs and enable segment routing.



CHAPTER 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 5500 Series Router, see the *Implementing IS-IS* module in the *Routing Configuration Guide for Cisco NCS 5500 Series Routers*.

- [Enabling Segment Routing for IS-IS Protocol, on page 41](#)
- [Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 43](#)
- [Configuring an Adjacency SID, on page 46](#)
- [Configuring Bandwidth-Based Local UCMP, on page 52](#)
- [IS-IS Multi-Domain Prefix SID and Domain Stitching: Example, on page 53](#)
- [Conditional Prefix Advertisement, on page 55](#)
- [Segment Routing ECMP-FEC Optimization, on page 57](#)

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**
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 global configuration mode.
Step 2	router isis <i>instance-id</i> Example: RP/0/RP0/CPU0:router(config)# router isis isp	Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode. 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 ipv4 unicast	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.
Step 4	metric-style wide [level { 1 2 }] Example: RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide level 1	Configures a router to generate and accept only wide link metrics in the Level 1 area.
Step 5	router-id loopback <i>loopback interface used for prefix-sid</i>	Configures router ID for each address-family (IPv4/IPv6).

	Command or Action	Purpose
	Example: <pre>RP/0/RP0/CPU0:router(config-isis-af)#router-id loopback0</pre>	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 Example: <pre>RP/0/RP0/CPU0:router(config-isis-af)# segment-routing mpls</pre>	Segment routing is enabled by the following actions: <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.
Step 7	exit Example: <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.	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. • 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.

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** [**algorithm** *algorithm-number*] {**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 global configuration 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.
Step 4	address-family { ipv4 ipv6 } [unicast] Example: The following is an example for ipv4 address family: RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

	Command or Action	Purpose
<p>Step 5</p>	<p>prefix-sid [<i>algorithm algorithm-number</i>] {<i>index SID-index</i> <i>absolute SID-value</i>} [<i>n-flag-clear</i>] [<i>explicit-null</i>]</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 absolute 17001</pre>	<p>Configures the prefix-SID index or absolute value for the interface.</p> <p>Specify algorithm <i>algorithm-number</i> to configure SR Flexible Algorithm.</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 <i>n-flag-clear</i> 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 <i>explicit-null</i> keyword. IS-IS sets the E flag in the prefix-SID sub TLV.</p>
<p>Step 6</p>	<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                      0/0/0
  Hostname:     router
  IP Address:   10.0.0.1
  IPv6 Address: 2001:0db8:1234::0a00:0001
  Router Cap:   10.0.0.1, D:0, S:0
    Segment Routing: I:1 V:1, SRGB Base: 16000 Range: 8000
    SR Algorithm:
      Algorithm: 0
```

<...>

```

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

<...>

```

Configuring an Adjacency SID

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

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

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

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

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

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

```

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

```

Table 1: Adjacency Segment Identifier (Adj-SID) Flags Sub-TLV Fields

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

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

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

Before you begin

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>configure</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# configure</pre>	Enters global configuration mode.
Step 2	<p>router isis <i>instance-id</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# router isis 1</pre>	<p>Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.</p> <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	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis)# interface GigabitEthernet0/0/0/7</pre>	Specifies the interface and enters interface configuration mode.
Step 4	<p>point-to-point</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis-if)# point-to-point</pre>	Specifies the interface is a point-to-point interface.
Step 5	<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>	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

	Command or Action	Purpose
Step 6	<p>adjacency-sid {<i>index</i> <i>adj-SID-index</i> absolute <i>adj-SID-value</i> } [protected]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# adjacency-sid index 10</pre> <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# 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 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 the Adj-SID configuration:

```
RP/0/RP0/CPU0:router# show isis segment-routing label adjacency persistent
Mon Jun 12 02:44:07.085 PDT

IS-IS 1 Manual Adjacency SID Table

15010 AF IPv4
  GigabitEthernet0/0/0/3: IPv4, Protected 1/65/N, Active
  GigabitEthernet0/0/0/7: IPv4, Protected 2/66/N, Active

15100 AF IPv6
  GigabitEthernet0/0/0/3: IPv6, Not protected 255/255/N, Active
```

Verify the labels are added to the MPLS Forwarding Information Base (LFIB):

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

```
16004          SRLB (idx 10)          Gi0/0/0/3          10.0.3.3          0          (!)
```

Manually Configure a Layer 2 Adjacency SID

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

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

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

- IGP dynamically allocates Layer 2 Adj-SIDs from the dynamic label range for each Layer 2 bundle member. A dynamic Layer 2 Adj-SID is not persistent and can be reallocated as the Layer 3 bundle link goes up and down.
- Manually configured Layer 2 Adj-SIDs are persistent if the Layer 3 bundle link goes up and down. Layer 2 Adj-SIDs are allocated from the Segment Routing Local Block (SRLB) range of labels. However, if the configured value of Layer 2 Adj-SID does not fall within the available SRLB, a Layer 2 Adj-SID will not be programmed into forwarding information base (FIB).

Restrictions

- Adj-SID forwarding requires a next-hop, which can be either an IPv4 address or an IPv6 address, but not both. Therefore, manually configured Layer 2 Adj-SIDs are configured per address-family.
- Manually configured Layer 2 Adj-SID can be associated with only one Layer 2 bundle member link.
- A SID value used for Layer 2 Adj-SID cannot be shared with Layer 3 Adj-SID.
- SR-TE using Layer 2 Adj-SID is not supported.

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

Before you begin

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

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

SUMMARY STEPS

1. **configure**
2. **segment-routing**
3. **adjacency-sid**
4. **interface** *type interface-path-id*
5. **address-family** { **ipv4** | **ipv6** } [**unicast**]
6. **l2-adjacency sid** {**index** *adj-SID-index* | **absolute** *adj-SID-value* } [**next-hop** { *ipv4_address* | *ipv6_address* }]
7. Use the **commit** or **end** command.
8. **end**

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters global configuration mode.
Step 2	segment-routing Example: Router(config)# <code>segment-routing</code>	Enters segment routing configuration mode.
Step 3	adjacency-sid Example: Router(config-sr)# <code>adjacency-sid</code>	Enters adjacency SID configuration mode.
Step 4	interface type interface-path-id Example: Router(config-sr-adj)# <code>interface GigabitEthernet0/0/0/3</code>	Specifies the interface and enters interface configuration mode.
Step 5	address-family { ipv4 ipv6 } [unicast] Example: Router(config-sr-adj-intf)# <code>address-family ipv4 unicast</code>	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.
Step 6	l2-adjacency sid {index adj-SID-index absolute adj-SID-value } [next-hop { ipv4_address ipv6_address }] Example: Router(config-sr-adj-intf-af)# <code>l2-adjacency sid absolute 15015 next-hop 10.1.1.4</code>	<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>For point-to-point interfaces, you are not required to specify a next-hop. However, if you do specify the next-hop, the Layer 2 Adj-SID will be used only if the specified next-hop matches the neighbor address.</p> <p>For LAN interfaces, you must configure the next-hop IPv4 or IPv6 address. If you do not configure the next-hop, the Layer 2 Adj-SID will not be used for LAN interface.</p>

	Command or Action	Purpose
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.
Step 8	end	
Step 9	<p>router isis <i>instance-id</i></p> <p>Example:</p> <pre>Router(config)# router isis isp</pre>	Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.
Step 10	<p>address-family { ipv4 ipv6 } [unicast]</p> <p>Example:</p> <pre>Router(config-isis)# address-family ipv4 unicast</pre>	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.
Step 11	<p>segment-routing bundle-member-adj-sid</p> <p>Example:</p> <pre>Router(config-isis-af) # segment-routing bundle-member-adj-sid</pre>	<p>Programs the dynamic Layer 2 Adj-SIDs, and advertises both manual and dynamic Layer 2 Adj-SIDs.</p> <p>Note This command is not required to program manual L2 Adj-SID, but is required to program the dynamic Layer 2 Adj-SIDs and to advertise both manual and dynamic Layer 2 Adj-SIDs.</p>

Verify the configuration:

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

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

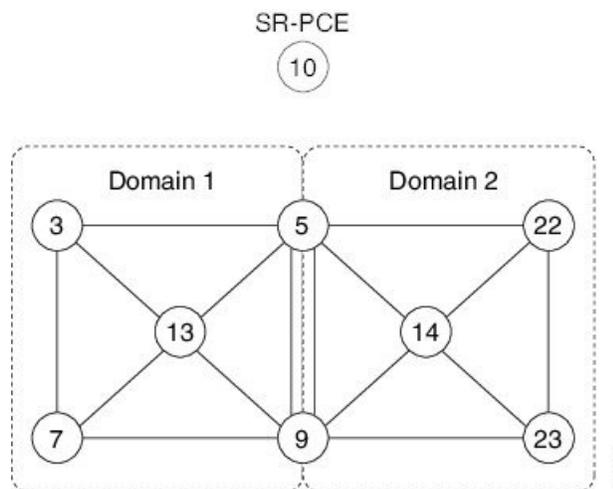

	Command or Action	Purpose
	RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast	
Step 4	<p>apply-weight ecmp-only bandwidth</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-isis-af)# apply-weight ecmp-only bandwidth</pre>	Enables UCMP functionality locally between ECMP paths based on the bandwidth of the local links.
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.

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 2: Multi-Domain Topology



Configure IS-IS Multi-Domain Prefix SID

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

Example: Border Node 5

```
router isis Domain1
 interface Loopback0
  address-family ipv4 unicast
  prefix-sid absolute 16005
```

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

Example: Border Node 9

```
router isis Domain1
 interface Loopback0
  address-family ipv4 unicast
  prefix-sid absolute 16009
```

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

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

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

Configure Common Router ID

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

Example: Border Node 5

```
router isis Domain1
 address-family ipv4 unicast
 router-id loopback0
```

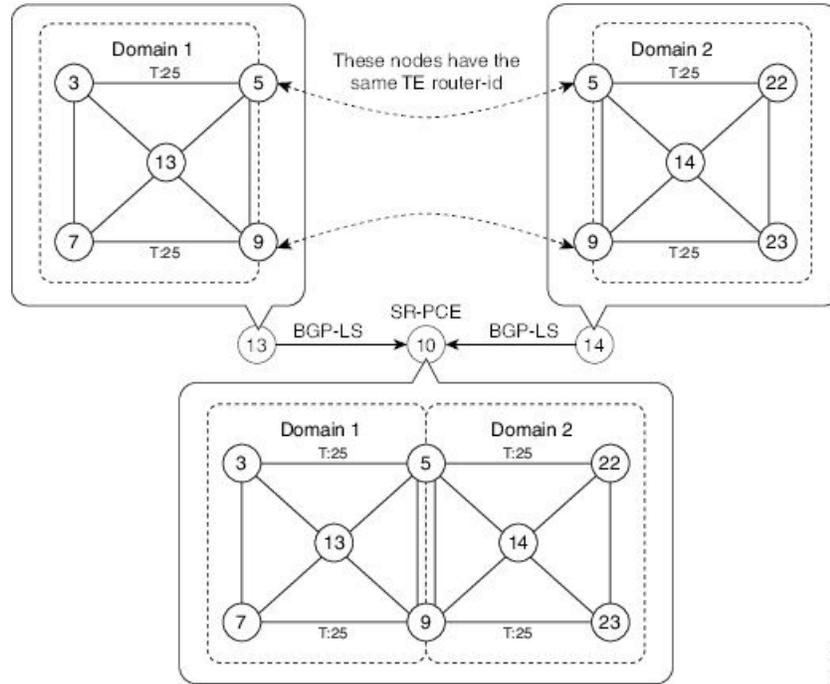
```
router isis Domain2
 address-family ipv4 unicast
 router-id loopback0
```

Example: Border Node 9

```
router isis Domain1
 address-family ipv4 unicast
 router-id loopback0
```

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

Distribute IS-IS Link-State Data



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

Example: Node 13

```
router isis Domain1
  distribute link-state instance-id instance-id
```

Example: Node 14

```
router isis Domain2
  distribute link-state instance-id instance-id
```

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

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

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

Conditional Prefix Advertisement

In some situations, it's beneficial to make the IS-IS prefix advertisement conditional. For example, an Area Border Router (ABR) or Autonomous System Boundary Router (ASBR) that has lost its connection to one of the areas or autonomous systems (AS) might keep advertising a prefix. If an ABR or ASBR advertises the Segment Routing (SR) SID with this prefix, the label stack of the traffic routed toward the disconnected area or AS might use this SID, which would result in dropped traffic at the ABR or ASBR.

ABRs or ASBRs are often deployed in pairs for redundancy and advertise a shared Anycast prefix SID. Conditional Prefix Advertisement allows an ABR or an ASBR to advertise its Anycast SID only when connected to a specific area or domain. If an ABR or ASBR becomes disconnected from the particular area or AS, it stops advertising the address for a specified interface (for example, Loopback).

Configure the conditional prefix advertisement under a specific interface. The prefix advertisement on this interface is associated with the route-policy that tracks the presence of a set of prefixes (prefix-set) in the Routing Information Base (RIB).

For faster convergence, the route-policy used for conditional prefix advertisement uses the new event-based **rib-has-route async** condition to notify IS-IS of the following situations:

- When the last prefix from the prefix-set is removed from the RIB.
- When the first prefix from the prefix-set is added to the RIB.

Configuration

To use the conditional prefix advertisement in IS-IS, create a prefix-set to be tracked. Then create a route policy that uses the prefix-set.

```
Router(config)# prefix-set prefix-set-name
Router(config-pxf)# prefix-address-1/length[, prefix-address-2/length,,
prefix-address-16/length]
Router(config-pxf)# end-set

Router(config)# route-policy rpl-name
Router(config-rpl)# if rib-has-route async prefix-set-name then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
```

To advertise the loopback address in IS-IS conditionally, use the **advertise prefix route-policy** command under IS-IS interface address-family configuration sub-mode.

```
Router(config)# router isis 1
Router(config-isis)# interface Loopback0
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# advertise prefix route-policy rpl-name
Router(config-isis-if-af)# commit
```

Example

```
Router(config)# prefix-set domain_2
Router(config-pxf)# 2.3.3.3/32, 2.4.4.4/32
Router(config-pxf)# end-set
Router(config)# route-policy track_domain_2
Router(config-rpl)# if rib-has-route async domain_2 then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
Router(config)# router isis 1
Router(config-isis)# interface Loopback0
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# advertise prefix route-policy track_domain-2
Router(config-isis-if-af)# commit
```

Running Configuration

```
prefix-set domain_2
 2.3.3.3/32,
 2.4.4.4/32
end-set
!
route-policy track_domain_2
  if rib-has-route async domain_2 then
    pass
  endif
end-policy
!
router isis 1
  interface Loopback0
    address-family ipv4 unicast
    advertise prefix route-policy track_domain_2
  !
  !
  !
```

Segment Routing ECMP-FEC Optimization

ECMP-FECs are used for any ECMP programming on the system, such as MPLS LSP ECMP, VPN multipath, and [EVPN multi-homing](#).

The SR ECMP-FEC optimization solution minimizes ECMP-FEC resource consumption during underlay programming for an SR-MPLS network. This feature supports sharing the same ECMP-FEC, regular FEC, and Egress Encapsulation DB (EEDB) entries among /32 IPv4 Segment Routing prefixes with the same set of next hops.

ECMP-FEC optimization is triggered when all the out_labels associated with the ECMP paths for a given prefix have the same value. If this rule is not met, then the prefix is programmed with a dedicated ECMP-FEC.

Segment Routing Label Edge Router (LER) ECMP-FEC Optimization enables ECMP-FEC optimization originally developed for Label Switched Router (LSR) nodes (MPLS P) to be enabled on LER (Layer 3 MPLS PE) routers.

Usage Guidelines and Limitations

- For IPv4 /32 labeled prefixes with ECMP across a combination of labeled and unlabeled (PHP) paths, the SR ECMP-FEC Optimization cannot be triggered since the paths associated with the prefix do not have the same outgoing label and/or label action.
- For prefixes with LFA backup paths, the SR ECMP-FEC Optimization is possible since these backup paths do not require an extra label to be pushed; all paths associated with the prefix (primary and backup) have the same outgoing label value.
- For prefixes with TI-LFA backup paths requiring extra labels to be pushed on to the backup, the SR ECMP-FEC Optimization is not possible since all the paths associated with the prefix do not have the same outgoing label value.
- For the duration of time that prefixes are programmed to avoid microloops (when SR MicroLoop Avoidance is triggered), SR ECMP-FEC Optimization is not possible since all the paths associated with the prefix do not have the same outgoing label value. After removal of the microloop-avoidance programming, the SR ECMP-FEC Optimization might be possible again.

- For scenarios with prefixes where the SR ECMP-FEC Optimization is not possible, dedicated ECMP-FEC is allocated per prefix. This could potentially lead to ECMP FEC out-of-resource (OOR) considering the baseline usage of ECMP FEC resources at steady state. During ECMP-FEC OOR, prefixes with multiple paths are programmed with a single path in order to avoid traffic disruption.
- SR ECMP-FEC optimization is applicable in the following instances:
 - Label Switched Router (LSR) nodes (MPLS P)
 - L3VPN Label Edge Router (LER) nodes
- SR ECMP-FEC optimization should not be enabled in the following instances:
 - L2VPN LER nodes
 - L2VPN/L3VPN LER nodes with VPN over BGP-LU over SR
- BGP PIC is not supported
- For IPv4 /32 labeled prefixes, transitioning from TI-LFA to SR ECMP-FEC optimization can cause ECMP-FEC OOR due to different output labels (ECMP label vs backup path's label) at make-before-break. This results in a few second traffic loss depending on route scale.

Enable SR ECMP-FEC Optimization

To enable SR ECMP-FEC optimization, use the **hw-module fib mpls label lsr-optimized** command in global configuration mode. After enabling this feature, reload the line card.

```
Router(config)# hw-module fib mpls label lsr-optimized
Router(config)# commit

LC/0/0/CPU0:Oct 11 20:19:12.540 UTC: fia_driver[185]:
%FABRIC-FIA_DRV-4-MPLS_HW_PROFILE_MISMATCH :
  Mismatch found, reload LC to activate the new mpls profile

Router# reload location 0/0/CPU0

Proceed with reload? [confirm]
Reloading node 0/0/CPU0
```

Verification

The following example shows NPU usage before enabling SR ECMP-FEC optimization.

```
Router# show controllers npu resources ecmpfec location all
HW Resource Information For Location: 0/0/CPU0
HW Resource Information
  Name                               : ecmp_fec

OOR Information
  NPU-0
    Estimated Max Entries             : 4096
    Red Threshold                     : 95
    Yellow Threshold                  : 80
    OOR State                         : Green

Current Usage
  NPU-0
    Total In-Use                      : 1001      (24 %)
```

```
ipnhgroup          : 1001      (24 %)  
ip6nhgroup         : 0         (0 %)
```

The following example shows NPU usage after enabling SR ECMP-FEC optimization.

```
Router# show controllers npu resources ecmpfec location all  
HW Resource Information For Location: 0/0/CPU0  
HW Resource Information  
  Name                : ecmp_fec  
  
OOR Information  
  NPU-0  
    Estimated Max Entries : 4096  
    Red Threshold         : 95  
    Yellow Threshold      : 80  
    OOR State             : Green  
  
Current Usage  
  NPU-0  
    Total In-Use         : 7      (0 %)  
    ipnhgroup            : 7      (0 %)  
    ip6nhgroup           : 0      (0 %)
```




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 5500 Series Router, see the *Implementing OSPF* module in the *Routing Configuration Guide for Cisco NCS 5500 Series Routers*.

- [Enabling Segment Routing for OSPF Protocol, on page 61](#)
- [Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 63](#)
- [Configuring an Adjacency SID, on page 65](#)
- [Segment Routing ECMP-FEC Optimization, on page 67](#)

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. **area** *area*
5. **segment-routing mpls**
6. **exit**
7. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration 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	segment-routing mpls Example: RP/0/RP0/CPU0:router(config-ospf)# segment-routing mpls	Enables segment routing using the MPLS data plane on the routing process and all areas and interfaces in the routing process. Enables segment routing forwarding on all interfaces in the routing process and installs the SIDs received by OSPF in the forwarding table.
Step 4	area <i>area</i> Example: RP/0/RP0/CPU0:router(config-ospf)# area 0	Enters area configuration mode.
Step 5	segment-routing mpls Example: RP/0/RP0/CPU0:router(config-ospf-ar)# segment-routing mpls	(Optional) Enables segment routing using the MPLS data plane on the area and all interfaces in the area. Enables segment routing forwarding on all interfaces in the area and installs the SIDs received by OSPF in the forwarding table.
Step 6	exit Example:	

	Command or Action	Purpose
	RP/0/RP0/CPU0:router(config-ospf-ar)# exit RP/0/RP0/CPU0:router(config-ospf)# exit	
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.

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** [**algorithm** *algorithm-number*] {**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 global configuration 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 [algorithm <i>algorithm-number</i>] { 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	Configures the prefix-SID index or absolute value for the interface. Specify algorithm <i>algorithm-number</i> to configure SR Flexible Algorithm. Specify index <i>SID-index</i> for each node to create a prefix SID based on the lower boundary of the SRGB + the index. Specify absolute <i>SID-value</i> for each node to create a specific prefix SID within the SRGB. By default, the n-flag is set on the prefix-SID, indicating that it is a node SID. For specific prefix-SID (for example, Anycast prefix-SID), enter the <code>n-flag-clear</code> keyword. OSPF does not set the N flag in the prefix-SID sub Type Length Value (TLV). To disable penultimate-hop-popping (PHP) and add an explicit-Null label, enter the <code>explicit-null</code> keyword. OSPF sets the E flag in the prefix-SID sub TLV.

	Command or Action	Purpose
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.

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|L|G|P|      |
+-----+-----+

```

Table 2: 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 global configuration 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 instance, and places the router in router configuration mode.

	Command or Action	Purpose
Step 3	area <i>area</i> Example: RP/0/RP0/CPU0:router(config-ospf)# area 0	Enters area configuration mode.
Step 4	interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-ospf-ar)# interface HundredGigE0/0/0/1	Specifies the interface and enters interface configuration mode.
Step 5	adjacency-sid { index <i>adj-SID-index</i> absolute <i>adj-SID-value</i> } [protected] Example: RP/0/RP0/CPU0:router(config-config-ospf-ar-if)# adjacency-sid index 10 RP/0/RP0/CPU0:router(config-config-ospf-ar-if)# adjacency-sid absolute 15010	Configures the Adj-SID index or absolute value for the interface. 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. Specify absolute <i>adj-SID-value</i> for each link to create a specific Adj-SID within the SRLB. Specify if the Adj-SID is protected . For each primary path, if the Adj-SID is protected on the primary interface and a backup path is available, a backup path is installed. By default, manual Adj-SIDs are not protected.
Step 6	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. • 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.

Segment Routing ECMP-FEC Optimization

ECMP-FECs are used for any ECMP programming on the system, such as MPLS LSP ECMP, VPN multipath, and [EVPN multi-homing](#).

The SR ECMP-FEC optimization solution minimizes ECMP-FEC resource consumption during underlay programming for an SR-MPLS network. This feature supports sharing the same ECMP-FEC, regular FEC, and Egress Encapsulation DB (EEDB) entries for all /32 IPv4 Segment Routing prefixes with the same set of

next hops. ECMP-FEC optimization is triggered when all the `out_labels` associated with the ECMP paths for a given prefix have the same value. If this rule is not met, then the prefix is programmed with a dedicated ECMP-FEC. Other prefixes that meet the rule are candidates for optimization.

Segment Routing Label Edge Router (LER) ECMP-FEC Optimization enables ECMP-FEC optimization originally developed for Label Switched Router (LSR) nodes (MPLS P) to be enabled on LER (Layer 3 MPLS PE) routers.

For usage guidelines, limitations, and configuration options, see [Segment Routing ECMP-FEC Optimization, on page 57](#).



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 router, see the *BGP Configuration Guide for Cisco NCS 5500 Series Routers*.

- [Segment Routing for BGP, on page 69](#)
- [Configure BGP Prefix Segment Identifiers, on page 70](#)
- [Segment Routing Egress Peer Engineering, on page 71](#)
- [Configure BGP Link-State, on page 73](#)
- [Use Case: Configuring SR-EPE and BGP-LS, on page 77](#)
- [Configure BGP Proxy Prefix SID, on page 79](#)

Segment Routing for BGP

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

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

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

Configure BGP Prefix Segment Identifiers

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

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



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

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



Note A routing policy with the **set label-index** attribute can be attached to a network configuration or redistribute configuration. Other routing policy language (RPL) configurations are possible. For more information on routing policies, refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco NCS 5500 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 <i>bgp 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. • 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
 !
 !
 !
```

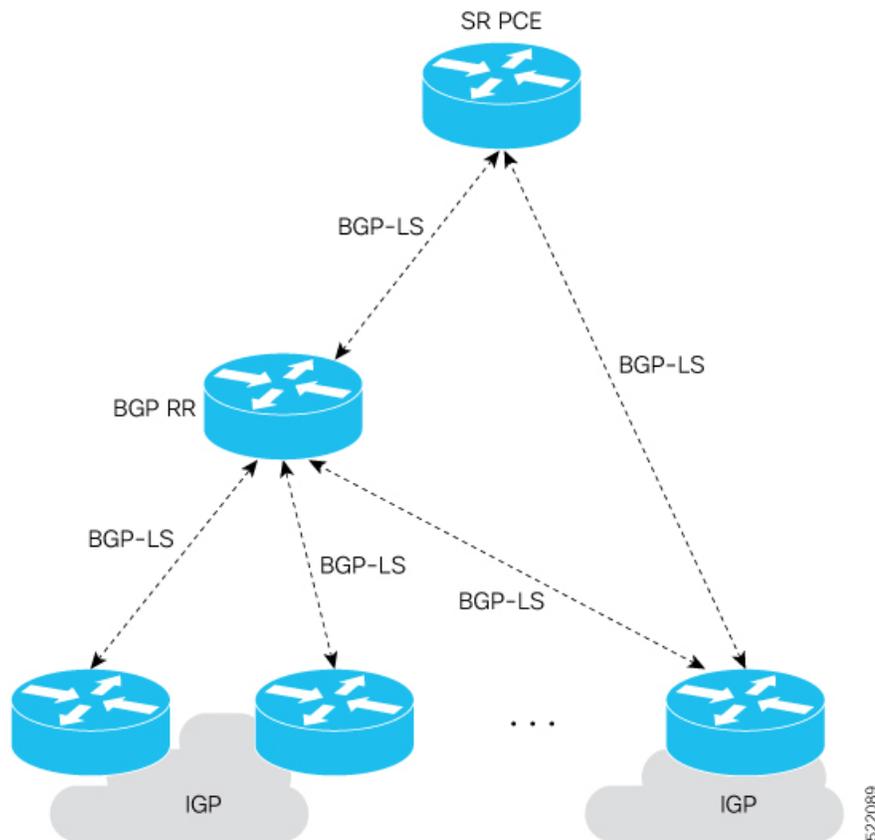
Configure BGP Link-State

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 NRIs 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.
- NRIs with different Instance-ID values are considered to be from different IGP routing instances.
- Unique Instance-ID values must be assigned to routing protocol instances operating in different IGP domains. This allows the BGP-LS Consumer (for example, SR-PCE) to build an accurate segregated multi-domain topology based on the Instance-ID values, even when the topology is advertised via BGP-LS by multiple BGP-LS Producers in the network.
- If the BGP-LS Instance-ID configuration guidelines are not followed, a BGP-LS Consumer may see duplicate link-state objects for the same node, link, or prefix when there are multiple BGP-LS Producers deployed. This may also result in the BGP-LS Consumers getting an inaccurate network-wide topology.

- The following table defines the supported extensions to the BGP-LS address family for carrying IGP topology information (including SR information) via BGP. For more information on the BGP-LS TLVs, refer to [Border Gateway Protocol - Link State \(BGP-LS\) Parameters](#).

Table 3: IOS XR Supported BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs

TLV Code Point	Description	Produced by IS-IS	Produced by OSPFv2	Produced by BGP
256	Local Node Descriptors	X	X	—
257	Remote Node Descriptors	X	X	—
258	Link Local/Remote Identifiers	X	X	—
259	IPv4 interface address	X	X	—
260	IPv4 neighbor address	X		
261	IPv6 interface address	X	—	—
262	IPv6 neighbor address	X	—	—
263	Multi-Topology ID	X	—	—
264	OSPF Route Type	—	X	—
265	IP Reachability Information	X	X	—
266	Node MSD TLV	X	X	—
267	Link MSD TLV	X	X	—
512	Autonomous System	—	—	X
513	BGP-LS Identifier	—	—	X
514	OSPF Area-ID	—	X	—
515	IGP Router-ID	X	X	—
516	BGP Router-ID TLV	—	—	X
517	BGP Confederation Member TLV	—	—	X
1024	Node Flag Bits	X	X	—
1026	Node Name	X	X	—
1027	IS-IS Area Identifier	X	—	—
1028	IPv4 Router-ID of Local Node	X	X	—
1029	IPv6 Router-ID of Local Node	X	—	—
1030	IPv4 Router-ID of Remote Node	X	X	—
1031	IPv6 Router-ID of Remote Node	X	—	—
1034	SR Capabilities TLV	X	X	—
1035	SR Algorithm TLV	X	X	—
1036	SR Local Block TLV	X	X	—

TLV Code Point	Description	Produced by IS-IS	Produced by OSPFv2	Produced by BGP
1039	Flex Algo Definition (FAD) TLV	X	X	—
1044	Flex Algorithm Prefix Metric (FAPM) TLV	X	X	—
1088	Administrative group (color)	X	X	—
1089	Maximum link bandwidth	X	X	—
1090	Max. reservable link bandwidth	X	X	—
1091	Unreserved bandwidth	X	X	—
1092	TE Default Metric	X	X	—
1093	Link Protection Type	X	X	—
1094	MPLS Protocol Mask	X	X	—
1095	IGP Metric	X	X	—
1096	Shared Risk Link Group	X	X	—
1099	Adjacency SID TLV	X	X	—
1100	LAN Adjacency SID TLV	X	X	—
1101	PeerNode SID TLV	—	—	X
1102	PeerAdj SID TLV	—	—	X
1103	PeerSet SID TLV	—	—	X
1114	Unidirectional Link Delay TLV	X	X	—
1115	Min/Max Unidirectional Link Delay TLV	X	X	—
1116	Unidirectional Delay Variation TLV	X	X	—
1117	Unidirectional Link Loss	X	X	—
1118	Unidirectional Residual Bandwidth	X	X	—
1119	Unidirectional Available Bandwidth	X	X	—
1120	Unidirectional Utilized Bandwidth	X	X	—
1122	Application-Specific Link Attribute TLV	X	X	—
1152	IGP Flags	X	X	—
1153	IGP Route Tag	X	X	—
1154	IGP Extended Route Tag	X	—	—
1155	Prefix Metric	X	X	—
1156	OSPF Forwarding Address	—	X	—
1158	Prefix-SID	X	X	—
1159	Range	X	X	—

TLV Code Point	Description	Produced by IS-IS	Produced by OSPFv2	Produced by BGP
1161	SID/Label TLV	X	X	—
1170	Prefix Attribute Flags	X	X	—
1171	Source Router Identifier	X	—	—
1172	L2 Bundle Member Attributes TLV	X	—	—
1173	Extended Administrative Group	X	X	—

Exchange Link State Information with BGP Neighbor

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

```
Router# configure
Router(config)# router bgp 1
Router(config-bgp)# neighbor 10.0.0.2
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# address-family link-state link-state
Router(config-bgp-nbr-af)# exit
```

IGP Link-State Database Distribution

A given BGP node may have connections to multiple, independent routing domains. IGP link-state database distribution into BGP-LS is supported for both OSPF and IS-IS protocols in order to distribute this information on to controllers or applications that desire to build paths spanning or including these multiple domains.

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

```
Router# configure
Router(config)# router isis isp
Router(config-isis)# distribute link-state instance-id 32
```

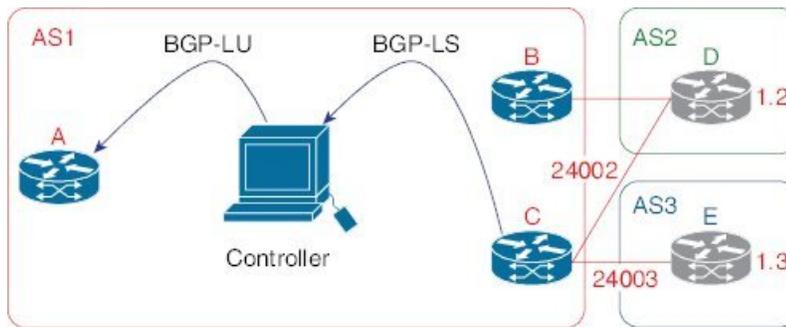
To distribute OSPFv2 link-state data using BGP-LS, use the **distribute link-state** command in router configuration mode.

```
Router# configure
Router(config)# router ospf 100
Router(config-ospf)# distribute link-state instance-id 32
```

Use Case: Configuring SR-EPE and BGP-LS

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

Figure 3: 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  -----
-----
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).

Configure BGP Proxy Prefix SID

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

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

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

Configuration Example:

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

```
RP/0/RSP0/CPU0:router(config)# segment-routing
RP/0/RSP0/CPU0:router(config-sr)# mapping-server
RP/0/RSP0/CPU0:router(config-sr-ms)# prefix-sid-map
RP/0/RSP0/CPU0:router(config-sr-ms-map)# address-family ipv4
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)# 10.1.1.1/32 10 range 200
RP/0/RSP0/CPU0:router(config-sr-ms-map-af)# 192.168.64.1/32 400 range 300
```

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

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

Verification

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

```
RP/0/RSP0/CPU0:router# show segment-routing mapping-server prefix-sid-map ipv4 detail
Prefix
10.1.1.1/32
  SID Index:      10
  Range:          200
  Last Prefix:    10.1.1.200/32
  Last SID Index: 209
  Flags:
Number of mapping entries: 1

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

BGP routing table entry for 192.168.64.1/32
Versions:
  Process          bRIB/RIB  SendTblVer
  Speaker          117      117
  Local Label: 16400
Last Modified: Oct 25 01:02:28.562 for 00:11:45Paths: (2 available, best #1)
Advertised to peers (in unique update groups):
  201.1.1.1
Path #1: Received by speaker 0 Advertised to peers (in unique update groups):
  201.1.1.1
  Local
  20.0.101.1 from 20.0.101.1 (20.0.101.1)      Received Label 61
  Origin IGP, localpref 100, valid, internal, best, group-best, multipath, labeled-unicast

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

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

```

Routing entry for 192.168.64.1/32
  Known via "bgp 65000", distance 200, metric 0, [ei]-bgp, labeled SR, type internal
  Installed Oct 25 01:02:28.583 for 00:20:09
  Routing Descriptor Blocks
    20.0.101.1, from 20.0.101.1, BGP multi path
      Route metric is 0
      Label: 0x3d (61)
      Tunnel ID: None
      Binding Label: None
      Extended communities count: 0
      NHID:0x0(Ref:0)
      Route version is 0x6 (6)
  Local Label: 0x3e81 (16400)
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type RIB_SVD_TYPE_LOCAL
  Download Priority 4, Download Version 242
  No advertising protos.

```

```

RP/0/RSP0/CPU0:router# show cef ipv4 192.168.64.1/32 detail
192.168.64.1/32, version 476, labeled SR, drop adjacency, internal 0x5000001 0x80 (ptr
0x71c42b40) [1], 0x0 (0x71c11590), 0x808 (0x722b91e0)
  Updated Oct 31 23:23:48.733
  Prefix Len 32, traffic index 0, precedence n/a, priority 4
  Extensions: context-label:16400
    gateway array (0x71ae7e78) reference count 3, flags 0x7a, source rib (7), 0 backups
      [2 type 5 flags 0x88401 (0x722eb450) ext 0x0 (0x0)]
    LW-LDI[type=5, refc=3, ptr=0x71c11590, sh-ldi=0x722eb450]
    gateway array update type-time 3 Oct 31 23:49:11.720
  LDI Update time Oct 31 23:23:48.733
  LW-LDI-TS Oct 31 23:23:48.733
    via 20.0.101.1/32, 0 dependencies, recursive, bgp-ext [flags 0x6020]
      path-idx 0 NHID 0x0 [0x7129a294 0x0]
      recursion-via-/32
      unresolved
      local label 16400
      labels imposed {ExpNullv6}

```

```

RP/0/RSP0/CPU0:router# show bgp labels
BGP router identifier 2.1.1.1, local AS number 65000
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000000 RD version: 245
BGP main routing table version 245
BGP NSR Initial initsync version 16 (Reached)
BGP NSR/ISSU Sync-Group versions 245/0
BGP scan interval 60 secs

```

```

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

```

Network	Next Hop	Rcvd Label	Local Label
*>i10.1.1.1/32	10.1.1.1	3	16010
*> 2.1.1.1/32	0.0.0.0	nolabel	3
*> 192.68.64.1/32	20.0.101.1	2	16400
*> 192.68.64.2/32	20.0.101.1	2	16401



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 83](#)
- [Usage Guidelines and Limitations, on page 84](#)
- [Instantiation of an SR Policy, on page 84](#)
- [SR-TE Policy Path Types, on page 115](#)
- [Protocols, on page 125](#)
- [Traffic Steering, on page 132](#)
- [Miscellaneous, on page 144](#)

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.

- 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.
- SR-TE over BVI is not supported. An SR-TE policy cannot be resolved over an MPLS-enabled BVI interface.
- **For NCS 5500:** Counter implications when BVI and SR-TE co-exist in same NPU—Counters for a BVI's logical interface are not allocated when the same NPU hosts layer-2 (sub)interface(s) associated with the BVI alongside other port(s) used as egress interface(s) for an SR policy
- GRE tunnel as primary interface for an SR policy is not supported.
- GRE tunnel as backup interface for an SR policy with TI-LFA protection is not supported.
- Head-end computed inter-domain SR policy with Flex Algo constraint and IGP redistribution is not supported.

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:

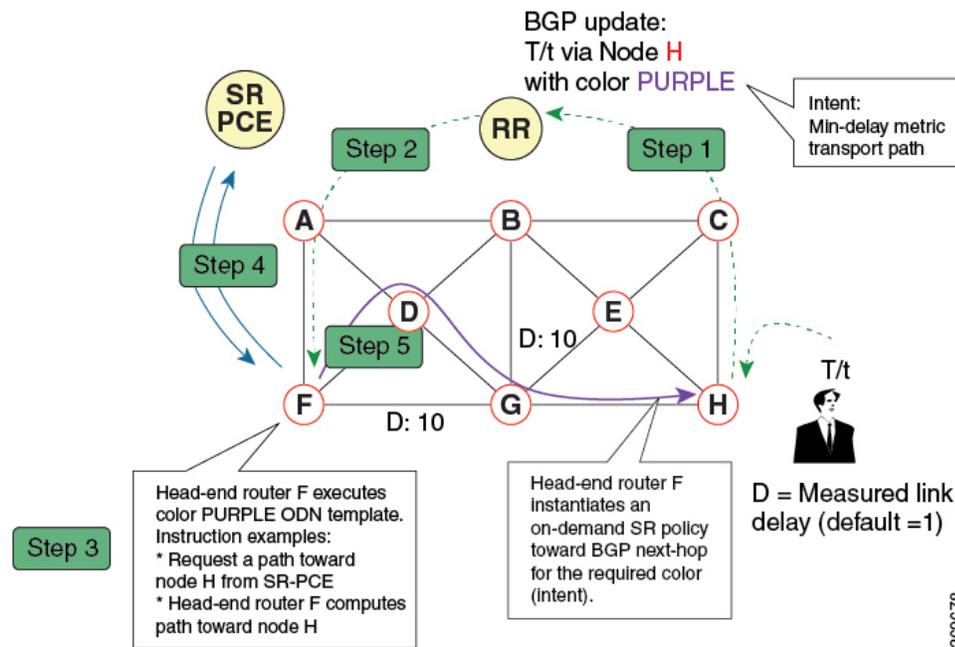
- [On-Demand SR Policy – SR On-Demand Next-Hop](#), on page 84
- [Manually Provisioned SR Policy](#), on page 115
- [PCE-Initiated SR Policy](#), on page 115

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 170](#).
 - b. Configure the head-end router as Path Computation Element Protocol (PCEP) Path Computation Client (PCC). For detailed PCEP PCC configuration information, see [Configure the Head-End Router as PCEP PCC](#).
2. Define BGP color extended communities. Refer to the "Implementing BGP" chapter in the [BGP Configuration Guide for NCS 5500 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 NCS 5500 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
EVI export	X	–
EVI import	X	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 NCS 5500 Series Routers](#).

Configure On-Demand Color Template

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

The *color* range is from 1 to 4294967295.

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



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

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

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

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

```
Router(config-sr-te)# on-demand color 10 dynamic pcep
```

Configure Dynamic Path Optimization Objectives

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

```
Router(config-sr-te-color-dyn)# metric type te
```

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

```
Router(config-sr-te-color-dyn)# metric margin absolute 5
```

Configure Dynamic Path Constraints

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

```
Router(config-sr-te-color-dyn)# disjoint-path group-id 775 type link
```

- Use the **affinity** {*include-any* | *include-all* | *exclude-any*} {*name WORD*} command to configure the affinity constraints.

```
Router(config-sr-te-color-dyn)# affinity exclude-any name CROSS
```

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

The default MSD *value* is equal to the maximum MSD supported by the platform (12).

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

See [Customize MSD Value at PCC, on page 127](#) 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
- color 30: minimization objective = delay-metric
- color 128: constraints = flex-algo

```
segment-routing
traffic-eng
on-demand color 10
dynamic
metric
type te
!
!
on-demand color 20
dynamic
metric
type igp
!
!
on-demand color 21
dynamic
metric
type igp
!
```

```

        affinity exclude-any
        name CROSS
    !
    !
    on-demand color 22
    dynamic
    pcep
    !
    metric
    type te
    !
    affinity exclude-any
    name CROSS
    !
    !
    !
    on-demand color 30
    dynamic
    metric
    type latency
    !
    !
    !
    on-demand color 128
    dynamic
    sid-algorithm 128
    !
    !
end

```

Configuring BGP Color Extended Community Set: Example

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



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

```

extcommunity-set opaque color10-te
  10
end-set
!
extcommunity-set opaque color20-igp
  20
end-set
!
extcommunity-set opaque color21-igp-excl-cross
  21
end-set
!
extcommunity-set opaque color30-delay
  30
end-set
!
extcommunity-set opaque color128-fa128
  128
end-set
!

```

Configuring RPL to Set BGP Color (Layer-3 Services): Examples

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

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

```
route-policy SET_COLOR_LOW_LATENCY_TE
  set extcommunity color color10-te
  pass
end-policy
!
route-policy SET_COLOR_HI_BW
  set extcommunity color color20-igp
  pass
end-policy
!
route-policy SET_COLOR_LOW_LATENCY
  set extcommunity color color30-delay
  pass
end-policy
!
route-policy SET_COLOR_FA_128
  set extcommunity color color128-fa128
  pass
end-policy
!

prefix-set sample-set
  88.1.0.0/24
end-set
!
route-policy SET_COLOR_GLOBAL
  if destination in sample-set then
    set extcommunity color color10-te
  else
    pass
  endif
end-policy
```

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

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

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

```
vrf vrf_cust1
  address-family ipv4 unicast
    export route-policy SET_COLOR_LOW_LATENCY_TE
  !
  address-family ipv6 unicast
    export route-policy SET_COLOR_LOW_LATENCY_TE
  !
!
vrf vrf_cust2
  address-family ipv4 unicast
    export route-policy SET_COLOR_HI_BW
  !
  address-family ipv6 unicast
    export route-policy SET_COLOR_HI_BW
```

```

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

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

router bgp 100
 neighbor-group BR-TO-RR
 address-family ipv4 unicast
   route-policy SET_COLOR_GLOBAL out
!
!
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 List	Outgoing Label	Outgoing Interface	Next Hop	Bytes Switched	Pure Backup
10	10.1.1.8	dynamic	16009	Gi0/0/0/4	10.4.5.5	0	
			16001	Gi0/0/0/5	11.4.8.8	0	Yes

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

```
Mon Jul 8 11:56:46.887 PST
```

```
SR-TE Policy Forwarding database
```

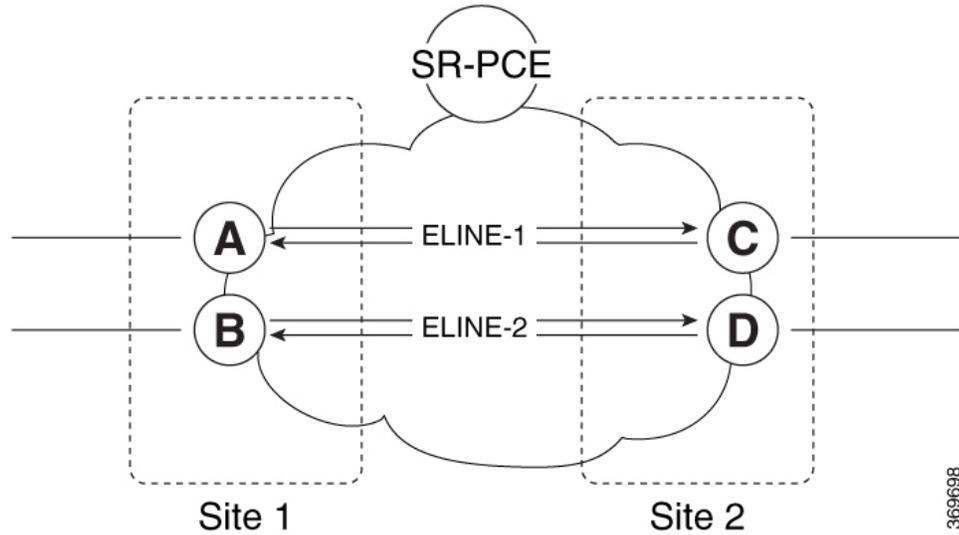
```
-----
Color: 10, End-point: 10.1.1.8
Name: srte_c_10_ep_10.1.1.8
Binding SID: 24036
Segment Lists:
  SL[0]:
    Name: dynamic
    Paths:
      Path[0]:
        Outgoing Label: 16009
        Outgoing Interface: GigabitEthernet0/0/0/4
        Next Hop: 10.4.5.5
        Switched Packets/Bytes: 0/0
        FRR Pure Backup: No
        Label Stack (Top -> Bottom): { 16009, 16008 }
        Path-id: 1 (Protected), Backup-path-id: 2, Weight: 64
      Path[1]:
        Outgoing Label: 16001
        Outgoing Interface: GigabitEthernet0/0/0/5
        Next Hop: 11.4.8.8
        Switched Packets/Bytes: 0/0
        FRR Pure Backup: Yes
        Label Stack (Top -> Bottom): { 16001, 16009, 16008 }
        Path-id: 2 (Pure-Backup), Weight: 64
Policy Packets/Bytes Switched: 0/0
Local label: 80013
```

Configuring SR-ODN 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 4: 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 96	Verification: SR-PCE, on page 100
Configuration: Site 1 Node A, on page 96	Verification: Site 1 Node A, on page 104
Configuration: Site 1 Node B, on page 97	Verification: Site 1 Node B, on page 107
Configuration: Site 2 Node C, on page 98	Verification: Site 2 Node C, on page 110
Configuration: Site 2 Node D, on page 99	Verification: Site 2 Node D, on page 112

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 172](#) for more information.

Configuration: Site 1 Node A

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

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

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

```

/* EVPN-VPWS configuration */

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

/* BGP color community and RPL configuration */

extcommunity-set opaque color-10000
 10000
end-set
!
route-policy SET_COLOR_EVPN_VPWS

```

```

    if evpn-route-type is 1 and rd in (ios-regex '.*..*..*..*:(100)') then
        set extcommunity color color-10000
    endif
    pass
end-policy
!
router bgp 65000
neighbor 10.1.1.253
address-family l2vpn evpn
route-policy SET_COLOR_EVPN_VPWS out
!
!
!

/* ODN template configuration */

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

```

Configuration: Site 1 Node B

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

```

/* EVPN-VPWS configuration */

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

/* BGP color community and RPL configuration */

extcommunity-set opaque color-10000
    10000
end-set
!
route-policy SET_COLOR_EVPN_VPWS
    if evpn-route-type is 1 and rd in (ios-regex '.*..*..*..*:(101)') then
        set extcommunity color color-10000
    endif
    pass
end-policy
!

```

```

router bgp 65000
 neighbor 10.1.1.253
   address-family l2vpn evpn
     route-policy SET_COLOR_EVPN_VPWS out
   !
 !
 !

/* ODN template configuration */

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

```

Configuration: Site 2 Node C

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

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

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

```

/* EVPN-VPWS configuration */

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

/* BGP color community and RPL configuration */

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

```

```

endif
pass
end-policy
!
router bgp 65000
neighbor 10.1.1.253
address-family l2vpn evpn
route-policy SET_COLOR_EVPN_VPWS out
!
!
!

/* ODN template configuration */

```

```

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

```

Configuration: Site 2 Node D

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

```

/* EVPN-VPWS configuration */

interface GigabitEthernet0/0/0/1.2500 l2transport
encapsulation dot1q 2500
rewrite ingress tag pop 1 symmetric
!
l2vpn
xconnect group evpn_vpws_group
p2p evpn_vpws_101
interface GigabitEthernet0/0/0/1.2500
neighbor evpn evi 101 target 12 source 22
!
!
!

/* BGP color community and RPL configuration */

extcommunity-set opaque color-11000
11000
end-set
!
route-policy SET_COLOR_EVPN_VPWS
if evpn-route-type is 1 and rd in (ios-regex '.*.*.*.*.*:(101)') then
set extcommunity color color-11000
endif
pass
end-policy
!
router bgp 65000
neighbor 10.1.1.253

```

```

address-family l2vpn evpn
  route-policy SET_COLOR_EVPN_VPWS out
!
!
!

/* ODN template configuration */

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

```

Verification: SR-PCE

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

```

RP/0/0/CPU0:SR-PCE# show pce ipv4 peer
Mon Jul 15 19:41:43.622 UTC

PCE's peer database:
-----
Peer address: 10.1.1.2
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 10.1.1.4
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 10.1.1.5
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

Peer address: 10.1.1.6
  State: Up
  Capabilities: Stateful, Segment-Routing, Update, Instantiation

```

Use the **show pce association group-id** command to display information for the pair of LSPs assigned to a given association group-id value.

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

- At Node A (10.1.1.5): LSP symbolic name = bgp_c_11000_ep_10.1.1.2_discr_100
- At Node B (10.1.1.6): LSP symbolic name = bgp_c_11000_ep_10.1.1.4_discr_100

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

```
RP/0/0/CPU0:SR-PCE# show pce association group-id 775
Thu Jul 11 03:52:20.770 UTC
```

```
PCE's association database:
```

```
-----
Association: Type Link-Disjoint, Group 775, Not Strict
```

```
Associated LSPs:
```

```
LSP[0]:
```

```
PCC 10.1.1.6, tunnel name bgp_c_11000_ep_10.1.1.4_discr_100, PLSP ID 18, tunnel ID 17,
LSP ID 3, Configured on PCC
```

```
LSP[1]:
```

```
PCC 10.1.1.5, tunnel name bgp_c_11000_ep_10.1.1.2_discr_100, PLSP ID 18, tunnel ID 18,
LSP ID 3, Configured on PCC
```

```
Status: Satisfied
```

Use the **show pce lsp** command to display detailed information of an LSP present in the PCE's LSP database. This output shows details for the LSP at Node A (10.1.1.5) that is used to carry traffic of EVPN VPWS EVI 100 towards node C (10.1.1.2).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.5 name bgp_c_11000_ep_10.1.1.2_discr_100
Thu Jul 11 03:58:45.903 UTC
```

```
PCE's tunnel database:
```

```
-----
PCC 10.1.1.5:
```

```
Tunnel Name: bgp_c_11000_ep_10.1.1.2_discr_100
```

```
Color: 11000
```

```
Interface Name: srte_c_11000_ep_10.1.1.2
```

```
LSPs:
```

```
LSP[0]:
```

```
source 10.1.1.5, destination 10.1.1.2, tunnel ID 18, LSP ID 3
```

```
State: Admin up, Operation up
```

```
Setup type: Segment Routing
```

```
Binding SID: 80037
```

```
Maximum SID Depth: 10
```

```
Absolute Metric Margin: 0
```

```
Relative Metric Margin: 0%
```

```
Preference: 100
```

```
Bandwidth: signaled 0 kbps, applied 0 kbps
```

```
PCEP information:
```

```
PLSP-ID 0x12, flags: D:1 S:0 R:0 A:1 O:1 C:0
```

```
LSP Role: Exclude LSP
```

```
State-sync PCE: None
```

```
PCC: 10.1.1.5
```

```
LSP is subdelegated to: None
```

```
Reported path:
```

```
Metric type: IGP, Accumulated Metric 40
```

```
SID[0]: Adj, Label 80003, Address: local 11.5.8.5 remote 11.5.8.8
```

```
SID[1]: Node, Label 16007, Address 10.1.1.7
```

```
SID[2]: Node, Label 16002, Address 10.1.1.2
```

```
Computed path: (Local PCE)
```

```
Computed Time: Thu Jul 11 03:49:48 UTC 2019 (00:08:58 ago)
```

```
Metric type: IGP, Accumulated Metric 40
```

```
SID[0]: Adj, Label 80003, Address: local 11.5.8.5 remote 11.5.8.8
```

```
SID[1]: Node, Label 16007, Address 10.1.1.7
```

```
SID[2]: Node, Label 16002, Address 10.1.1.2
```

```
Recorded path:
```

```
None
```

```
Disjoint Group Information:
```

```
Type Link-Disjoint, Group 775
```

This output shows details for the LSP at Node B (10.1.1.6) that is used to carry traffic of EVPN VPWS EVI 101 towards node D (10.1.1.4).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.6 name bgp_c_11000_ep_10.1.1.4_discr_100
Thu Jul 11 03:58:56.812 UTC
```

```
PCE's tunnel database:
```

```
-----
PCC 10.1.1.6:
```

```
Tunnel Name: bgp_c_11000_ep_10.1.1.4_discr_100
```

```
Color: 11000
```

```
Interface Name: srte_c_11000_ep_10.1.1.4
```

```
LSPs:
```

```
LSP[0]:
```

```
source 10.1.1.6, destination 10.1.1.4, tunnel ID 17, LSP ID 3
```

```
State: Admin up, Operation up
```

```
Setup type: Segment Routing
```

```
Binding SID: 80061
```

```
Maximum SID Depth: 10
```

```
Absolute Metric Margin: 0
```

```
Relative Metric Margin: 0%
```

```
Preference: 100
```

```
Bandwidth: signaled 0 kbps, applied 0 kbps
```

```
PCEP information:
```

```
PLSP-ID 0x12, flags: D:1 S:0 R:0 A:1 O:1 C:0
```

```
LSP Role: Disjoint LSP
```

```
State-sync PCE: None
```

```
PCC: 10.1.1.6
```

```
LSP is subdelegated to: None
```

```
Reported path:
```

```
Metric type: IGP, Accumulated Metric 40
```

```
SID[0]: Node, Label 16001, Address 10.1.1.1
```

```
SID[1]: Node, Label 16004, Address 10.1.1.4
```

```
Computed path: (Local PCE)
```

```
Computed Time: Thu Jul 11 03:49:48 UTC 2019 (00:09:08 ago)
```

```
Metric type: IGP, Accumulated Metric 40
```

```
SID[0]: Node, Label 16001, Address 10.1.1.1
```

```
SID[1]: Node, Label 16004, Address 10.1.1.4
```

```
Recorded path:
```

```
None
```

```
Disjoint Group Information:
```

```
Type Link-Disjoint, Group 775
```

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

- At Node C (10.1.1.2): LSP symbolic name = bgp_c_10000_ep_10.1.1.5_discr_100
- At Node D (10.1.1.4): LSP symbolic name = bgp_c_10000_ep_10.1.1.6_discr_100

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

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

```
PCE's association database:
```

```
-----
Association: Type Link-Disjoint, Group 776, Not Strict
```

```
Associated LSPs:
```

```
LSP[0]:
```

```
PCC 10.1.1.4, tunnel name bgp_c_10000_ep_10.1.1.6_discr_100, PLSP ID 16, tunnel ID 14, LSP ID 1, Configured on PCC
```

```
LSP[1]:
```

PCC 10.1.1.2, tunnel name **bgp_c_10000_ep_10.1.1.5_discr_100**, PLSP ID 6, tunnel ID 21, LSP ID 3, Configured on PCC
Status: Satisfied

Use the **show pce lsp** command to display detailed information of an LSP present in the PCE's LSP database. This output shows details for the LSP at Node C (10.1.1.2) that is used to carry traffic of EVPN VPWS EVI 100 towards node A (10.1.1.5).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.2 name bgp_c_10000_ep_10.1.1.5_discr_100
Thu Jul 11 03:55:21.706 UTC
```

PCE's tunnel database:

PCC 10.1.1.2:

Tunnel Name: **bgp_c_10000_ep_10.1.1.5_discr_100**

Color: 10000

Interface Name: srte_c_10000_ep_10.1.1.5

LSPs:

LSP[0]:

source 10.1.1.2, destination 10.1.1.5, tunnel ID 21, LSP ID 3

State: Admin up, Operation up

Setup type: Segment Routing

Binding SID: 80052

Maximum SID Depth: 10

Absolute Metric Margin: 0

Relative Metric Margin: 0%

Preference: 100

Bandwidth: signaled 0 kbps, applied 0 kbps

PCEP information:

PLSP-ID 0x6, flags: D:1 S:0 R:0 A:1 O:1 C:0

LSP Role: Exclude LSP

State-sync PCE: None

PCC: 10.1.1.2

LSP is subdelegated to: None

Reported path:

Metric type: IGP, Accumulated Metric 40

SID[0]: Node, Label 16007, Address 10.1.1.7

SID[1]: Node, Label 16008, Address 10.1.1.8

SID[2]: Adj, Label 80005, Address: local 11.5.8.8 remote 11.5.8.5

Computed path: (Local PCE)

Computed Time: Thu Jul 11 03:50:03 UTC 2019 (00:05:18 ago)

Metric type: IGP, Accumulated Metric 40

SID[0]: Node, Label 16007, Address 10.1.1.7

SID[1]: Node, Label 16008, Address 10.1.1.8

SID[2]: Adj, Label 80005, Address: local 11.5.8.8 remote 11.5.8.5

Recorded path:

None

Disjoint Group Information:

Type Link-Disjoint, Group 776

This output shows details for the LSP at Node D (10.1.1.4) used to carry traffic of EVPN VPWS EVI 101 towards node B (10.1.1.6).

```
RP/0/0/CPU0:SR-PCE# show pce lsp pcc ipv4 10.1.1.4 name bgp_c_10000_ep_10.1.1.6_discr_100
Thu Jul 11 03:55:23.296 UTC
```

PCE's tunnel database:

PCC 10.1.1.4:

Tunnel Name: **bgp_c_10000_ep_10.1.1.6_discr_100**

Color: 10000

Interface Name: srte_c_10000_ep_10.1.1.6

```

LSPs:
LSP[0]:
  source 10.1.1.4, destination 10.1.1.6, tunnel ID 14, LSP ID 1
  State: Admin up, Operation up
  Setup type: Segment Routing
  Binding SID: 80047
  Maximum SID Depth: 10
  Absolute Metric Margin: 0
  Relative Metric Margin: 0%
  Preference: 100
  Bandwidth: signaled 0 kbps, applied 0 kbps
  PCEP information:
    PLSP-ID 0x10, flags: D:1 S:0 R:0 A:1 O:1 C:0
  LSP Role: Disjoint LSP
  State-sync PCE: None
  PCC: 10.1.1.4
  LSP is subdelegated to: None
  Reported path:
    Metric type: IGP, Accumulated Metric 40
    SID[0]: Node, Label 16001, Address 10.1.1.1
    SID[1]: Node, Label 16006, Address 10.1.1.6
  Computed path: (Local PCE)
    Computed Time: Thu Jul 11 03:50:03 UTC 2019 (00:05:20 ago)
    Metric type: IGP, Accumulated Metric 40
    SID[0]: Node, Label 16001, Address 10.1.1.1
    SID[1]: Node, Label 16006, Address 10.1.1.6
  Recorded path:
    None
  Disjoint Group Information:
    Type Link-Disjoint, Group 776

```

Verification: Site 1 Node A

This section depicts verification steps at Node A.

Use the **show bgp l2vpn evpn** command to display BGP prefix information for EVPN-VPWS EVI 100 (rd 10.1.1.5:100). The output includes an EVPN route-type 1 route with color 11000 originated at Node C (10.1.1.2).

```

RP/0/RSP0/CPU0:Node-A# show bgp l2vpn evpn rd 10.1.1.5:100
Wed Jul 10 18:57:57.704 PST
BGP router identifier 10.1.1.5, local AS number 65000
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0   RD version: 0
BGP main routing table version 360
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
               i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop          Metric LocPrf Weight Path
Route Distinguisher: 10.1.1.5:100 (default for vrf VPWS:100)
*> [1][0000.0000.0000.0000.0000][11]/120
      0.0.0.0                                0 i
*>i [1][0000.0000.0000.0000.0000][21]/120
      10.1.1.2 C:11000                        100    0 i

```

The following output displays the details for the incoming EVPN RT1. Note the presence of BGP extended color community 11000, and that the prefix is associated with an SR policy with color 11000 and BSID value of 80044.

```
RP/0/RSP0/CPU0:Node-A# show bgp l2vpn evpn rd 10.1.1.5:100
[1] [0000.0000.0000.0000.0000][21]/120
Wed Jul 10 18:57:58.107 PST
BGP routing table entry for [1][0000.0000.0000.0000.0000][21]/120, Route Distinguisher:
10.1.1.5:100
Versions:
  Process          bRIB/RIB  SendTblVer
  Speaker          360      360
Last Modified: Jul 10 18:36:18.369 for 00:21:40
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    10.1.1.2 C:11000 (bsid:80044) (metric 40) from 10.1.1.253 (10.1.1.2)
      Received Label 80056
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate,
imported, rib-install
      Received Path ID 0, Local Path ID 1, version 358
      Extended community: Color:11000 RT:65000:100
      Originator: 10.1.1.2, Cluster list: 10.1.1.253
      SR policy color 11000, up, registered, bsid 80044, if-handle 0x00001b20

      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 10.1.1.2:100
```

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

```
RP/0/RSP0/CPU0:Node-A# show l2vpn xconnect group evpn_vpws_group
Wed Jul 10 18:58:02.333 PST
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect          Segment 1          Segment 2
Group  Name      ST  Description      ST  Description      ST
-----
evpn_vpws_group
      evpn_vpws_100
      UP  Gi0/0/0/3.2500  UP  EVPN 100,21,10.1.1.2  UP
-----
```

The following output shows the details for the service. Note that the service is associated with the on-demand SR policy with color 11000 and end-point 10.1.1.2 (node C).

```
RP/0/RSP0/CPU0:Node-A# show l2vpn xconnect group evpn_vpws_group xc-name evpn_vpws_100
detail
Wed Jul 10 18:58:02.755 PST

Group evpn_vpws_group, XC evpn_vpws_100, state is up; Interworking none
AC: GigabitEthernet0/0/0/3.2500, state is up
Type VLAN; Num Ranges: 1
Rewrite Tags: []
VLAN ranges: [2500, 2500]
MTU 1500; XC ID 0x120000c; interworking none
Statistics:
  packets: received 0, sent 0
  bytes: received 0, sent 0
  drops: illegal VLAN 0, illegal length 0
```

```

EVPN: neighbor 10.1.1.2, PW ID: evi 100, ac-id 21, state is up ( established )
XC ID 0xa0000007
Encapsulation MPLS
Source address 10.1.1.5
Encap type Ethernet, control word enabled
Sequencing not set
Preferred path Active : SR TE srte_c_11000_ep_10.1.1.2, On-Demand, fallback enabled
Tunnel : Up
Load Balance Hashing: src-dst-mac

```

EVPN	Local	Remote
Label	80040	80056
MTU	1500	1500
Control word	enabled	enabled
AC ID	11	21
EVPN type	Ethernet	Ethernet

```

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

```

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

The following output shows the on-demand SR policy with BSID 80044 that was triggered by EVPN RT1 prefix with color 11000 advertised by node C (10.1.1.2).

```

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

```

Color	Endpoint	Admin 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 Group	Name	ST	Segment 1 Description	ST	Segment 2 Description	ST
evpn_vpws_group	evpn_vpws_101	UP	TenGigE0/3/0/0/8.2500	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 115](#) 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 174](#) 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 candidate path has the following characteristics:

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

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



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

Dynamic Paths

Behaviors and Limitations

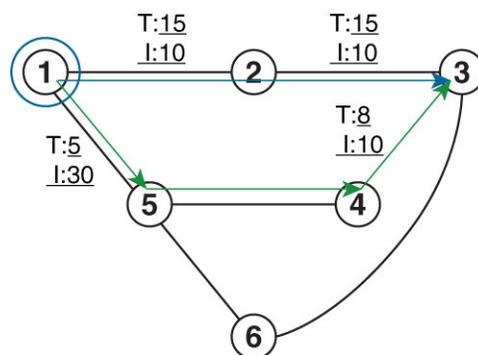
For a dynamic path that traverses a specific interface between nodes (segment), the algorithm may encode this segment using an Adj-SID. The SR-TE process prefers the protected Adj-SID of the link, if one is available.

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 Cisco NCS 5500 Series Routers*.
- TE metric — See the [Configure Interface TE Metrics, on page 117](#) 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

520016

- 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 117](#) 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 256 color names for affinity and attribute-flag attributes instead of 32-bit hexadecimal numbers. After mappings are defined, the attributes can be referred to by the corresponding color name in the CLI. Furthermore, you can define constraints using *include-any*, *include-all*, and *exclude-any* arguments, where each statement can contain up to 10 colors.



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

Configure Named Interface Link Admin Groups and SR-TE Affinity Maps

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

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# interface TenGigE0/0/1/2
Router(config-sr-if)# affinity
Router(config-sr-if-affinity)# name RED
```

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

Configure affinity maps on the following routers:

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

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# affinity-map
Router(config-sr-te-affinity-map)# name RED bit-position 23
```

Configuring Link Admin Group: Example

The following example shows how to assign affinity to interfaces and to define affinity maps. This configuration is applicable to any router (SR-TE head-end or transit node) with colored interfaces.

```
segment-routing
traffic-eng
interface TenGigE0/0/1/1
affinity
name CROSS
name RED
!
!
interface TenGigE0/0/1/2
affinity
name RED
!
```

```

!
interface TenGigE0/0/2/0
  affinity
    name BLUE
!
!
affinity-map
  name RED bit-position 23
  name BLUE bit-position 24
  name CROSS bit-position 25
!
end

```

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 116](#) section.
2. Define the constraints. See the [Constraints, on page 117](#) section.
3. Create the policy.

Behaviors and Limitations

For a dynamic path that traverses a specific interface between nodes (segment), the algorithm may encode this segment using an Adj-SID. The SR-TE process prefers the protected Adj-SID of the link, if one is available.

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.

Behaviors and Limitations

- An IP-defined segment can be associated with an IPv4 address (for example, a link or a Loopback address).
- When a segment of the segment list is defined as an MPLS label, subsequent segments can only be configured as MPLS labels.
- When configuring an explicit path using IP addresses of links along the path, the SR-TE process prefers the protected Adj-SID of the link, if one is available.

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-policy-path-pref)# 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-policy-path-pref)# exit
Router(config-sr-te-policy-path-pref)# exit
```



```

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-policy-path-pref)# exit
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST2
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path-pref)# 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 (12).

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



Note The platform's SR-TE label imposition capabilities are as follows:

- Up to 12 transport labels when no service labels are imposed
 - Up to 9 transport labels when service labels are imposed
-

For cases with path computation at PCE, a PCC can signal its MSD to the PCE in the following ways:

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



Note If the configured MSD values are different, the per-LSP MSD takes precedence over the per-node MSD.

After path computation, the resulting label stack size is verified against the MSD requirement.

- If the label stack size is larger than the MSD and path computation is performed by PCE, then the PCE returns a "no path" response to the PCC.
- If the label stack size is larger than the MSD and path computation is performed by PCC, then the PCC will not install the path.



Note A sub-optimal path (if one exists) that satisfies the MSD constraint could be computed in the following cases:

- For a dynamic path with TE metric, when the PCE is configured with the **pce segment-routing te-latency** command or the PCC is configured with the **segment-routing traffic-eng te-latency** command.
- For a dynamic path with LATENCY metric
- For a dynamic path with affinity constraints

For example, if the PCC MSD is 4 and the optimal path (with an accumulated metric of 100) requires 5 labels, but a sub-optimal path exists (with accumulated metric of 110) requiring 4 labels, then the sub-optimal path is installed.

Customize the SR-TE Path Calculation

Use the **te-latency** command to enable ECMP-aware path computation for TE metric.

```
Router(config-sr-te) # te-latency
```



Note ECMP-aware path computation is enabled by default for IGP and LATENCY metrics.

Configure PCEP Redundancy Type

Use the **redundancy pcc-centric** command to enable PCC-centric high-availability model. The PCC-centric model changes the default PCC delegation behavior to the following:

- After LSP creation, LSP is automatically delegated to the PCE that computed it.
- If this PCE is disconnected, then the LSP is redelegated to another PCE.
- If the original PCE is reconnected, then the delegation fallback timer is started. When the timer expires, the LSP is redelegated back to the original PCE, even if it has worse preference than the current PCE.

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

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

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

- Enable the SR-TE head-end router as a PCEP client (PCC) with 3 PCEP servers (PCE) with different precedence values. The PCE with IP address 10.1.1.57 is selected as BEST.
- Enable SR-TE related syslogs.
- Set the Maximum SID Depth (MSD) signaled during PCEP session establishment to 5.
- Enable PCEP reporting for all policies in the node.

```
segment-routing
traffic-eng
pcc
  source-address ipv4 10.1.1.2
  pce address ipv4 10.1.1.57
  precedence 150
  password clear <password>
  !
  pce address ipv4 10.1.1.58
  precedence 200
  password clear <password>
  !
  pce address ipv4 10.1.1.59
  precedence 250
  password clear <password>
  !
  !
  logging
  policy status
  !
  maximum-sid-depth 5
  pcc
  report-all
  !
  !
end
```

Verification

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

```
PCC's peer database:
```

```
-----
```

```
Peer address: 10.1.1.57, Precedence: 150, (best PCE)
```

```
State up
```

```
Capabilities: Stateful, Update, Segment-Routing, Instantiation
```

```
Peer address: 10.1.1.58, Precedence: 200
```

```
State up
```

```
Capabilities: Stateful, Update, Segment-Routing, Instantiation
```

```
Peer address: 10.1.1.59, Precedence: 250
```

```
State up
```

```
Capabilities: Stateful, Update, Segment-Routing, Instantiation
```

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
- IPv4 SR policy advertised over BGPv6 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	Configures the local router with a specified router ID.

	Command or Action	Purpose
	10.1.1.1	
Step 4	address-family {ipv4 ipv6} sr-policy Example: <pre>RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 sr-policy</pre>	Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.
Step 5	exit	
Step 6	neighbor ip-address Example: <pre>RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.0.1</pre>	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.
Step 7	remote-as as-number Example: <pre>RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 1</pre>	Creates a neighbor and assigns a remote autonomous system number to it.
Step 8	address-family {ipv4 ipv6} sr-policy Example: <pre>RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 sr-policy</pre>	Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.
Step 9	route-policy route-policy-name {in out} Example: <pre>RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass out</pre>	Applies the specified policy to IPv4 or IPv6 unicast routes.

Example: BGP SR-TE with BGPv4 Neighbor to BGP SR-TE Controller

The following configuration shows the an SR-TE head-end with a BGPv4 session towards a BGP SR-TE controller. This BGP session is used to signal both IPv4 and IPv6 SR policies.

```
router bgp 65000
bgp router-id 10.1.1.1
!
address-family ipv4 sr-policy
!
address-family ipv6 sr-policy
!
neighbor 10.1.3.1
  remote-as 10
```

```

description *** eBGP session to BGP SRTE controller ***
address-family ipv4 sr-policy
  route-policy pass in
  route-policy pass out
!
address-family ipv6 sr-policy
  route-policy pass in
  route-policy pass out
!
!
!
!

```

Example: BGP SR-TE with BGPv6 Neighbor to BGP SR-TE Controller

The following configuration shows an SR-TE head-end with a BGPv6 session towards a BGP SR-TE controller. This BGP session is used to signal both IPv4 and IPv6 SR policies.

```

router bgp 65000
bgp router-id 10.1.1.1
  address-family ipv4 sr-policy
  !
  address-family ipv6 sr-policy
  !
neighbor 3001::10:1:3:1
  remote-as 10
  description *** eBGP session to BGP SRTE controller ***
  address-family ipv4 sr-policy
    route-policy pass in
    route-policy pass out
  !
  address-family ipv6 sr-policy
    route-policy pass in
    route-policy pass out
  !
!
!
!

```

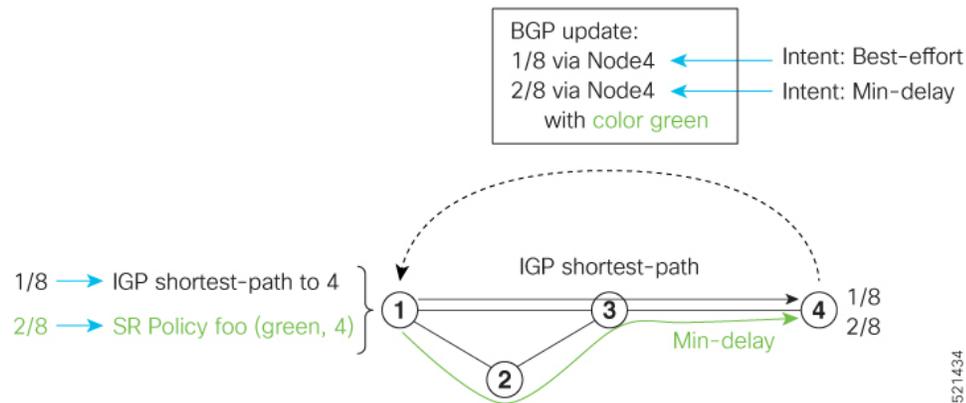
Traffic Steering

Automated Steering

Automated steering (AS) allows service traffic to be automatically steered onto the required transport SLA path programmed by an SR policy.

With AS, BGP automatically steers traffic onto an SR Policy based on the next-hop and color of a BGP service route. The color of a BGP service route is specified by a color extended community attribute. This color is used as a transport SLA indicator, such as min-delay or min-cost.

When the next-hop and color of a BGP service route matches the end-point and color of an SR Policy, BGP automatically installs the route resolving onto the BSID of the matching SR Policy. Recall that an SR Policy on a head-end is uniquely identified by an end-point and color.



When a BGP route has multiple extended-color communities, each with a valid SR Policy, the BGP process installs the route on the SR Policy giving preference to the color with the highest numerical value.

The granularity of AS behaviors can be applied at multiple levels, for example:

- At a service level—When traffic destined to all prefixes in a given service is associated to the same transport path type. All prefixes share the same color.
- At a destination/prefix level—When traffic destined to a prefix in a given service is associated to a specific transport path type. Each prefix could be assigned a different color.
- At a flow level—When flows destined to the same prefix are associated with different transport path types

AS behaviors apply regardless of the instantiation method of the SR policy, including:

- On-demand SR policy
- Manually provisioned SR policy
- PCE-initiated SR policy

See the [Verifying BGP VRF Information, on page 91](#) and [Verifying Forwarding \(CEF\) Table, on page 92](#) sections for sample output that shows AS implementation.

Color-Only Automated Steering

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

You can create an SR-TE policy for a specific color that uses a NULL end-point (0.0.0.0 for IPv4 NULL, and ::0 for IPv6 NULL end-point). This means that you can have a single policy that can steer traffic that is based on that color and a NULL endpoint for routes with a particular color extended community, but different destinations (next-hop).



Note Every SR-TE policy with a NULL end-point must have an explicit path-option. The policy cannot have a dynamic path-option (where the path is computed by the head-end or PCE) since there is no destination for the policy.

You can also specify a color-only (CO) flag in the color extended community for overlay routes. The CO flag allows the selection of an SR-policy with a matching color, regardless of endpoint Sub-address Family Identifier (SAFI) (IPv4 or IPv6). See [Setting CO Flag, on page 134](#).

Configure Color-Only Steering

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

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

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

Setting CO Flag

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



Note See [Color-Only Automated Steering, on page 133](#) for information about color-only steering (NULL end-point).

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

co-flag 00	<ol style="list-style-type: none"> 1. The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>. 2. If a policy does not exist, then IGP path for the next-hop N is chosen.
-------------------	--

co-flag 01	<ol style="list-style-type: none"> 1. The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>. 2. If a policy does not exist, then an SR-TE policy with NULL end-point with the same address-family as N and color C is chosen. 3. If a policy with NULL end-point with same address-family as N does not exist, then an SR-TE policy with any NULL end-point and color C is chosen. 4. If no match is found, then IGP path for the next-hop N is chosen.
-------------------	--

Configuration Example

```

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

```

Address-Family Agnostic Automated Steering

Address-family agnostic steering uses an SR-TE policy to steer both labeled and unlabeled IPv4 and IPv6 traffic. This feature requires support of IPv6 encapsulation (IPv6 caps) over IPV4 endpoint policy.

IPv6 caps for IPv4 NULL end-point is enabled automatically when the policy is created in Segment Routing Path Computation Element (SR-PCE). The binding SID (BSID) state notification for each policy contains an "ipv6_caps" flag that notifies SR-PCE clients (PCC) of the status of IPv6 caps (enabled or disabled).

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

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

Disable IPv6 Encapsulation

```

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

```

Using Binding Segments

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



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

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

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

BSID can be used in the following cases:

- Multi-Domain (inter-domain, inter-autonomous system)—BSIDs can be used to steer traffic across domain borders, creating seamless end-to-end inter-domain SR-TE policies.
- Large-Scale within a single domain—The head-end can use hierarchical SR-TE policies by nesting the end-to-end (edge-to-edge) SR-TE policy within another layer of SR-TE policies (aggregation-to-aggregation). The SR-TE policies are nested within another layer of policies using the BSIDs, resulting in seamless end-to-end SR-TE policies.
- Label stack compression—If the label-stack size required for an SR-TE policy exceeds the platform capability, the SR-TE policy can be seamlessly stitched to, or nested within, other SR-TE policies using a binding segment.

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 5: Stitching SR-TE Polices Using Binding SID

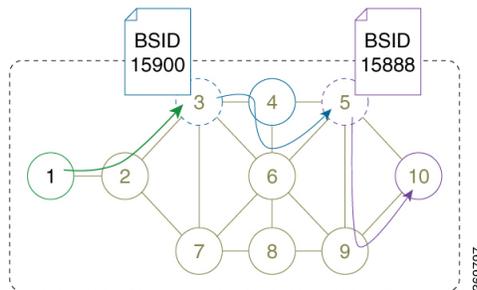


Table 5: Router IP Address

Router	Prefix Address	Prefix SID/Adj-SID
3	Loopback0 - 10.1.1.3	Prefix SID - 16003
4	Loopback0 - 10.1.1.4 Link node 4 to node 6 - 10.4.6.4	Prefix SID - 16004 Adjacency SID - dynamic
5	Loopback0 - 10.1.1.5	Prefix SID - 16005
6	Loopback0 - 10.1.1.6 Link node 4 to node 6 - 10.4.6.6	Prefix SID - 16006 Adjacency SID - dynamic
9	Loopback0 - 10.1.1.9	Prefix SID - 16009
10	Loopback0 - 10.1.1.10	Prefix SID - 16010

Step 1 On node 5, do the following:

- a) Define an SR-TE policy with an explicit path configured using the loopback interface IP addresses of node 9 and node 10.
- b) Define an explicit binding-SID (**mpls label 15888**) allocated from SRLB for the SR-TE policy.

Example:

Node 5

```
segment-routing
traffic-eng
segment-list PATH-9_10
index 10 address ipv4 10.1.1.9
index 20 address ipv4 10.1.1.10
!
policy foo
binding-sid mpls 15888
color 777 end-point ipv4 10.1.1.10
candidate-paths
preference 100
explicit segment-list PATH5-9_10
!
!
!
!
!
```

```
RP/0/RSP0/CPU0:Node-5# show segment-routing traffic-eng policy color 777
```

```
SR-TE policy database
-----
```

```
Color: 777, End-point: 10.1.1.10
Name: srte_c_777_ep_10.1.1.10
Status:
Admin: up Operational: up for 00:00:52 (since Aug 19 07:40:12.662)
Candidate-paths:
Preference: 100 (configuration) (active)
Name: foo
Requested BSID: 15888
PCC info:
Symbolic name: cfg_foo_discr_100
PLSP-ID: 70
Explicit: segment-list PATH-9_10 (valid)
Weight: 1, Metric Type: TE
16009 [Prefix-SID, 10.1.1.9]
16010 [Prefix-SID, 10.1.1.10]
Attributes:
Binding SID: 15888 (SRLB)
Forward Class: 0
Steering BGP disabled: no
IPv6 caps enable: yes
```

Step 2 On node 3, do the following:

- a) Define an SR-TE policy with an explicit path configured using the following:
 - Loopback interface IP address of node 4
 - Interface IP address of link between node 4 and node 6

- Loopback interface IP address of node 5
- Binding-SID of the SR-TE policy defined in Step 1 (**mpls label 15888**)

Note This last segment allows the stitching of these policies.

- b) Define an explicit binding-SID (**mpls label 15900**) allocated from SRLB for the SR-TE policy.

Example:

Node 3

```
segment-routing
traffic-eng
segment-list PATH-4_4-6_5_BSID
index 10 address ipv4 10.1.1.4
index 20 address ipv4 10.4.6.6
index 30 address ipv4 10.1.1.5
index 40 mpls label 15888
!
policy baa
binding-sid mpls 15900
color 777 end-point ipv4 10.1.1.5
candidate-paths
preference 100
explicit segment-list PATH-4_4-6_5_BSID
!
!
!
!
!
```

```
RP/0/RSP0/CPU0:Node-3# show segment-routing traffic-eng policy color 777
```

```
SR-TE policy database
-----
```

```
Color: 777, End-point: 10.1.1.5
Name: srte_c_777_ep_10.1.1.5
Status:
  Admin: up Operational: up for 00:00:32 (since Aug 19 07:40:32.662)
Candidate-paths:
  Preference: 100 (configuration) (active)
  Name: baa
  Requested BSID: 15900
  PCC info:
    Symbolic name: cfg_baa_discr_100
    PLSP-ID: 70
  Explicit: segment-list PATH-4_4-6_5_BSID (valid)
    Weight: 1, Metric Type: TE
    16004 [Prefix-SID, 10.1.1.4]
    80005 [Adjacency-SID, 10.4.6.4 - 10.4.6.6]
    16005 [Prefix-SID, 10.1.1.5]
    15888
Attributes:
  Binding SID: 15900 (SRLB)
  Forward Class: 0
  Steering BGP disabled: no
  IPv6 caps enable: yes
```

Step 3 On node 1, define an SR-TE policy with an explicit path configured using the loopback interface IP address of node 3 and the binding-SID of the SR-TE policy defined in step 2 (**mpls label 15900**). This last segment allows the stitching of these policies.

Example:

Node 1

```
segment-routing
traffic-eng
segment-list PATH-3_BSID
index 10 address ipv4 10.1.1.3
index 20 mpls label 15900
!
policy bar
color 777 end-point ipv4 10.1.1.3
candidate-paths
preference 100
explicit segment-list PATH-3_BSID
!
!
!
!
!
```

```
RP/0/RSP0/CPU0:Node-1# show segment-routing traffic-eng policy color 777
```

```
SR-TE policy database
```

```
-----
Color: 777, End-point: 10.1.1.3
Name: srte_c_777_ep_10.1.1.3
Status:
  Admin: up Operational: up for 00:00:12 (since Aug 19 07:40:52.662)
Candidate-paths:
  Preference: 100 (configuration) (active)
  Name: bar
  Requested BSID: dynamic
  PCC info:
    Symbolic name: cfg_bar_discr_100
    PLSP-ID: 70
  Explicit: segment-list PATH-3_BSID (valid)
    Weight: 1, Metric Type: TE
    16003 [Prefix-SID, 10.1.1.3]
    15900
Attributes:
  Binding SID: 80021
  Forward Class: 0
  Steering BGP disabled: no
  IPv6 caps enable: yes
```

L2VPN Preferred Path

EVPN VPWS Preferred Path over SR-TE Policy feature allows you to set the preferred path between the two end-points for EVPN VPWS pseudowire (PW) using SR-TE policy.

L2VPN VPLS or VPWS Preferred Path over SR-TE Policy feature allows you to set the preferred path between the two end-points for L2VPN Virtual Private LAN Service (VPLS) or Virtual Private Wire Service (VPWS) using SR-TE policy.

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*.

Static Route over Segment Routing Policy

This feature allows you to specify a Segment Routing (SR) policy as an interface type when configuring static routes for MPLS data planes.

For information on configuring static routes, see the "Implementing Static Routes" chapter in the *Routing Configuration Guide for Cisco NCS 5500 Series Routers*.

Configuration Example

The following example depicts a configuration of a static route for an IPv4 destination over an SR policy according to following parameters:

- Target SR policy:
 - Color = 200
 - End-point = 10.1.1.4
 - Auto-generated SR policy name = srte_c_200_ep_10.1.1.4



Note Use the auto-generated SR-TE policy name to attach the SR policy to the static route. Auto-generated SR policy names use the following naming convention: **srte_c_color_val_ep_endpoint-address**.

Use the `show segment-routing traffic-eng policy color <color_val> endpoint ipv4 <ip_addr>` command to display the auto-generated policy name.

- Admin distance = 40
- Load metric = 150
- Install the route in RIB regardless of reachability

```
Router(config)# router static
Router(config-static)# address-family ipv4 unicast
Router(config-static-afi)# 10.1.1.4/32 sr-policy srte_c_200_ep_10.1.1.4 40 permanent metric
150
```

Running Configuration

```
router static
address-family ipv4 unicast
  10.1.1.4/32 sr-policy srte_c_200_ep_10.1.1.4 40 permanent metric 150
!
```

!

Verification

```
RP/0/RP0/CPU0:RTR-1# show run segment-routing traffic-eng policy sample-policy-foo
```

```
Tue Feb 16 17:40:16.759 PST
```

```
segment-routing
traffic-eng
policy sample-policy-foo
  color 200 end-point ipv4 10.1.1.4
  candidate-paths
  preference 100
  dynamic
  metric
  type te
  !
  !
  !
  !
  !
  !
  !
```

```
RP/0/RP0/CPU0:RTR-1# show segment-routing traffic-eng policy color 200 endpoint ipv4 10.1.1.4
```

```
Tue Feb 16 17:17:45.724 PST
```

```
SR-TE policy database
```

```
-----
```

```
Color: 200, End-point: 10.1.1.4
```

```
Name: srte_c_200_ep_10.1.1.4
```

```
Status:
```

```
Admin: up Operational: up for 5d04h (since Feb 11 12:22:59.054)
```

```
Candidate-paths:
```

```
Preference: 100 (configuration) (active)
```

```
Name: sample-policy-foo
```

```
Requested BSID: dynamic
```

```
Protection Type: protected-preferred
```

```
Maximum SID Depth: 10
```

```
Dynamic (valid)
```

```
Metric Type: TE, Path Accumulated Metric: 14
```

```
16005 [Prefix-SID, 10.1.1.5]
```

```
16004 [Prefix-SID, 10.1.1.4]
```

```
Attributes:
```

```
Binding SID: 24014
```

```
Forward Class: Not Configured
```

```
Steering labeled-services disabled: no
```

```
Steering BGP disabled: no
```

```
IPv6 caps enable: yes
```

```
Invalidation drop enabled: no
```

```
RP/0/RP0/CPU0:RTR-1# show static sr-policy srte_c_200_ep_10.1.1.4
```

```
Tue Feb 16 17:50:19.932 PST
```

```
Interface          VRF          State      Paths
srte_c_200_ep_10.1.1.4  default      Up         10.1.1.4/32
Reference Count(in path with both intf<-->NH):0
Last IM notification was Up at Feb 16 17:09:08.325
```

```
Global ifh          : 0x0000007c
```

```
IM state            : up
```

```
RSI registration    : Yes
```

```
Table IDs           : 0xe0000000
```

```
Address Info:
10.1.1.1/32
Route tag: 0x00000000 Flags: 0x00000000 Prefix SID: False [Active]
```

```
IP-STATIC-IDB-CLASS
Total entries : 1
Interface      : sr-srte_c_200_ep_10.1.1.4
| Event Name   | Time Stamp           | S, M
| idb-create   | Feb 16 17:09:08.352 | 0, 0
```

```
RP/0/RP0/CPU0:RTR-1# show route 10.1.1.4/32
Tue Feb 16 17:09:21.164 PST
```

```
Routing entry for 10.1.1.4/32
  Known via "static", distance 40, metric 0 (connected)
  Installed Feb 16 17:09:08.325 for 00:00:13
  Routing Descriptor Blocks
    directly connected, via srte_c_200_ep_10.1.1.4, permanent
    Route metric is 0, Wt is 150
  No advertising protos.
```

```
RP/0/RP0/CPU0:RTR-1# show route 10.1.1.4/32 detail
Tue Feb 16 17:09:36.718 PST
```

```
Routing entry for 10.1.1.4/32
  Known via "static", distance 40, metric 0 (connected)
  Installed Feb 16 17:09:08.325 for 00:00:28
  Routing Descriptor Blocks
    directly connected, via srte_c_200_ep_10.1.1.4, permanent
    Route metric is 0, Wt is 150
    Label: None
    Tunnel ID: None
    Binding Label: None
    Extended communities count: 0
    NHID:0x0(Ref:0)
  Route version is 0x4a (74)
  Local Label: 0x3e84 (16004)
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_RECURSIVE (9) SVD Type RIB_SVD_TYPE_LOCAL
  Download Priority 3, Download Version 258
  No advertising protos.
```

```
RP/0/RP0/CPU0:RTR-1# show cef 10.1.1.4/32 detail
```

```
Tue Feb 16 17:10:06.956 PST
10.1.1.4/32, version 258, attached, internal 0x1000441 0x30 (ptr 0xd3f0d30) [1], 0x0
(0xe46f960), 0xa20 (0xe9694e0)
Updated Feb 16 17:09:08.328
Prefix Len 32, traffic index 0, precedence n/a, priority 3
gateway array (0xe2d9a08) reference count 2, flags 0x8068, source rib (7), 0 backups
[3 type 4 flags 0x108401 (0xe9aeb98) ext 0x0 (0x0)]
LW-LDI[type=1, refc=1, ptr=0xe46f960, sh-ldi=0xe9aeb98]
gateway array update type-time 1 Feb 16 17:07:59.946
LDI Update time Feb 16 17:07:59.946
LW-LDI-TS Feb 16 17:07:59.946
via srte_c_200_ep_10.1.1.4, 5 dependencies, weight 0, class 0 [flags 0xc]
path-idx 0 NHID 0x0 [0xf3b1a30 0x0]
local adjacency
local label 16004 labels imposed {None}
```

```
Load distribution: 0 (refcount 3)

Hash OK Interface Address
0 Y srte_c_200_ep_10.1.1.4 point2point
```

```
RP/0/RP0/CPU0:RTR-1# show mpls forwarding labels 16004 detail
Tue Feb 16 17:27:59.831 PST
Local Outgoing Prefix Outgoing Next Hop Bytes
Label Label or ID Interface Interface Switched
-----
16004 Unlabelled SR Pfx (idx 4) srte_c_200_e point2point 990
Updated: Feb 16 17:07:59.945
Path Flags: 0xc [ ]
Version: 258, Priority: 3
Label Stack (Top -> Bottom): { Unlabelled Unlabelled }
NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx: 0, Weight: 0
MAC/Encaps: 0/0, MTU: 0
Outgoing Interface: srte_c_200_ep_10.1.1.4 (ifhandle 0x0000007c)
Packets Switched: 20
```

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)
```

SR-TE MPLS Label Imposition Enhancement

The SR-TE MPLS Label Imposition Enhancement feature increases the maximum label imposition capabilities of the platform.

In previous releases, the platform supported:

- Up to 5 MPLS transport labels when no MPLS service labels are imposed
- Up to 3 MPLS transport labels when MPLS service labels are imposed

With the SR-TE MPLS Label Imposition Enhancement feature, the platform supports the following:

- Up to 12 MPLS transport labels when no MPLS service labels are imposed
- Up to 9 MPLS transport labels when MPLS service labels are imposed

This enhancement is enabled and disabled dynamically, as the label count changes. For example, if a path requires only 3 MPLS transport labels, the MPLS Label Imposition Enhancement feature is not enabled.

You can disable labeled services for SR-TE policies. The label switching database (LSD) needs to know if labeled services are disabled on top of an SR-TE policy to perform proper label stack splitting.

Disable Labeled Services per Local Policy

Use the **labeled-services disable** command to disable steering for labeled services for a configured policy. This configuration applies per policy.

```
segment-routing
 traffic-eng
  policy policy name
  steering
    labeled-services disable
```

Disable Labeled Services per ODN color

Use the **labeled-services disable** command to disable steering of labeled-services for on-demand color policies. This configuration applies for a specific ODN color.

```
segment-routing
 traffic-eng
  on-demand color color
  steering
    labeled-services disable
```

Disable Labeled Services per Policy Type

Use the **labeled-services disable** command to disable steering of labeled services for all policies for the following policy types:

- **all** — all policies
- **local** — all locally configured policies
- **on-demand** — all BGP on-demand color policies
- **bgp-srte** — all controller-initiated BGP SR-TE policies
- **pcep** — all PCE-initiated policies



Note You can specify more than one policy type.

```
segment-routing
 traffic-eng
  steering
    labeled-services
      disable {all | local | on-demand | bgp-srte | pcep}
```

Verification

Use the **show segment-routing traffic-eng policy** command to display SR policy information. The following output shows that steering of labeled services for the on-demand SR policy are disabled.

```
Router# show segment-routing traffic-eng policy color 10
Thu Jul 18 11:35:25.124 PDT

SR-TE policy database
-----

Color: 10, End-point: 10.1.1.8
Name: srte_c_10_ep_10.1.1.8
```

```

Status:
  Admin: up   Operational: up for 00:00:06 (since Jul 18 11:35:19.350)
Candidate-paths:
  Preference: 1 (configuration) (active)
    Name: test_pol_2
    Requested BSID: dynamic
    Dynamic (valid)
      Metric Type: TE,   Path Accumulated Metric: 10
      24004 [Adjacency-SID, 10.1.1.1 - 10.1.1.2]
Attributes:
  Binding SID: 24011
  Forward Class: 0
  Steering labeled-services disabled: yes
  Steering BGP disabled: no
  IPv6 caps enable: yes

```

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

Segment Routing Tree Segment Identifier

Tree Segment Identifier (Tree-SID) is a tree-building solution that uses a Segment Routing Path Computation Element (SR-PCE) using path computation element protocol (PCEP) to calculate the point-to-multipoint (P2MP) tree using SR policies. Tree-SID uses a single MPLS label for building a multicast replication tree in an SR network. Tree-SID does not require multicast control protocols such as RSVP, mLDP, and PIM.

A P2MP SR policy provides an SR-based TE solution for transporting multicast traffic. It works on existing data-plane (MPLS and IP) and supports TE capabilities and single/multi routing domains. At each node of the tree, the forwarding state is represented by the same segment (using a global Tree-SID specified from the SRLB range of labels). P2MP SR policy prevents transient loop and packet loss when updating the path of a P2MP SR policy.

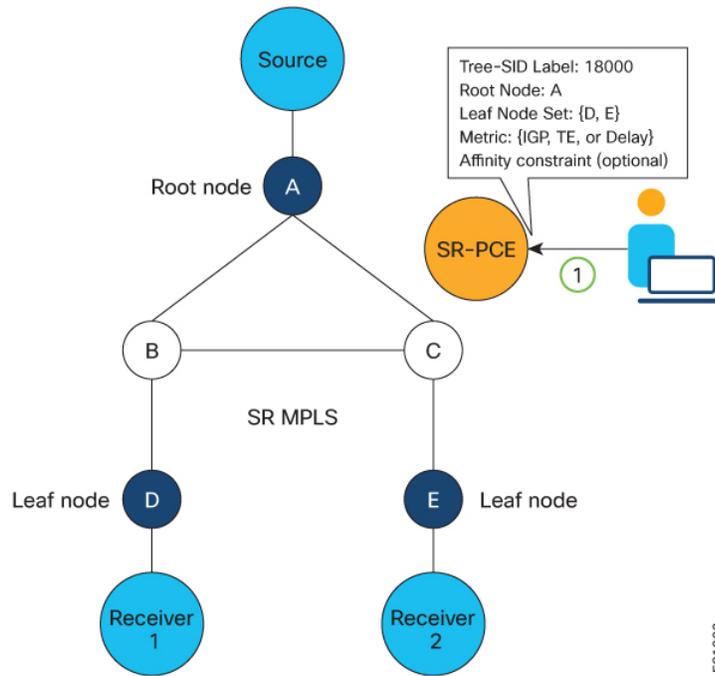
A P2MP SR policy request contains the following:

- Policy name
- SID for the P2MP Tree (Tree-SID)
- Address of the root node
- Addresses of the leaf nodes
- Optimization objectives (TE, IGP, delay metric)
- Constraints (affinity)

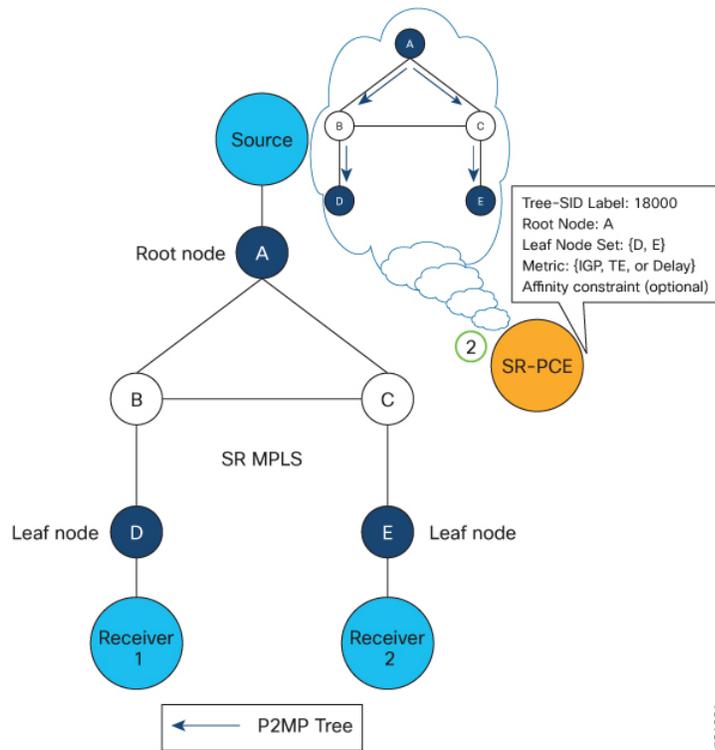
Tree SID Workflow Overview

This sections shows a basic workflow using a Tree SID policy:

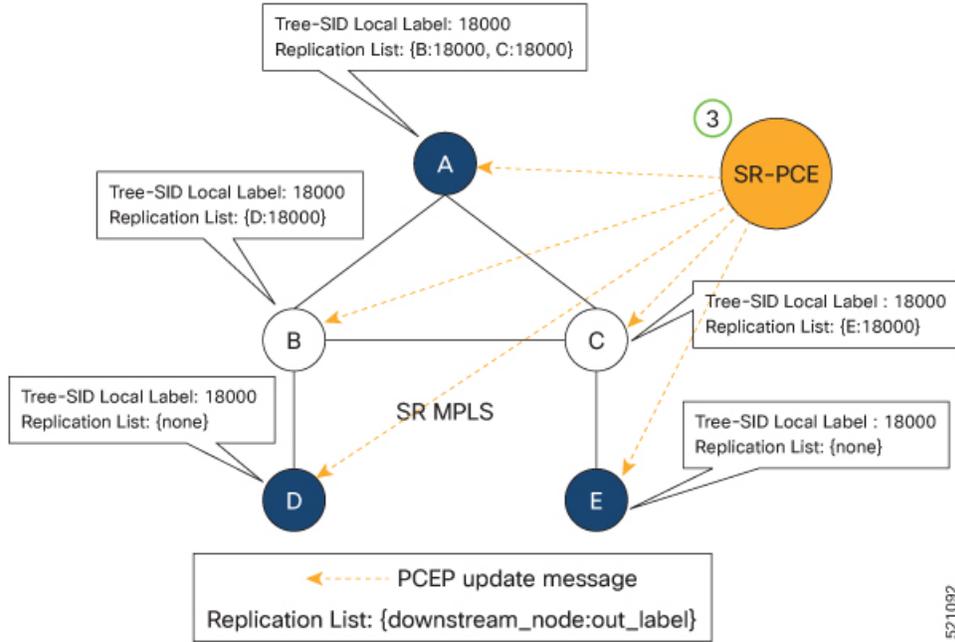
1. User creates a Tree-SID policy.



2. SR-PCE computes the P2MP Tree.

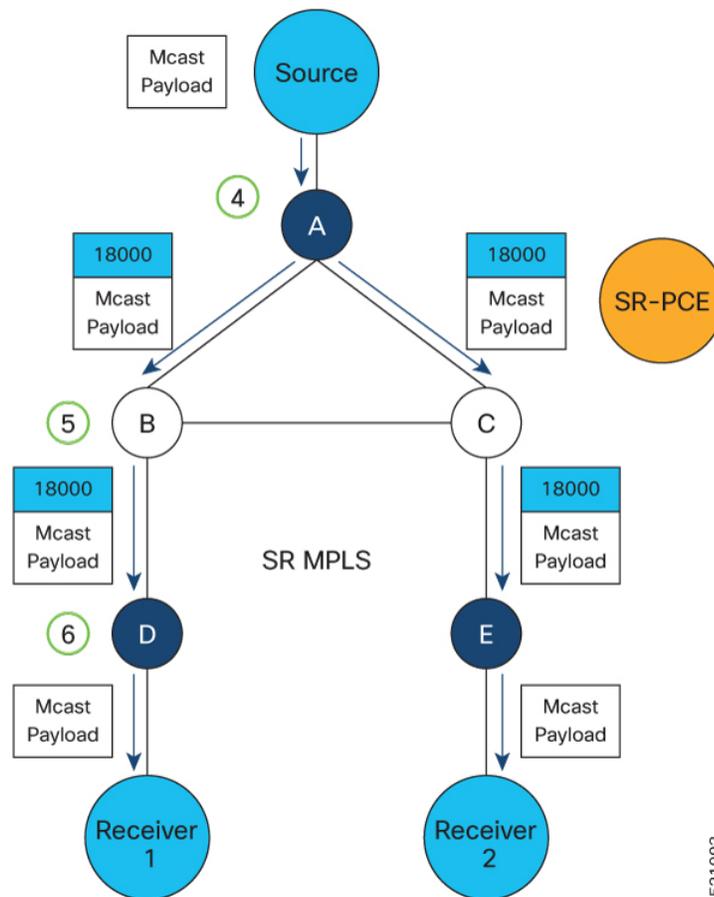


- SR-PCE instantiates the Tree-SID state at each node in the tree.



521092

- The Root node encapsulates the multicast traffic, replicates it, and forwards it to the Transit nodes.
- The Transit nodes replicate the multicast traffic and forward it to the Leaf nodes.
- The Leaf nodes decapsulate the multicast traffic and forward it to the multicast receivers.



- [Behaviors and Limitations](#), on page 154
- [Bud Node Support](#), on page 154
- [Configure Segment Routing Tree-SID](#), on page 155
- [Running Config](#), on page 157

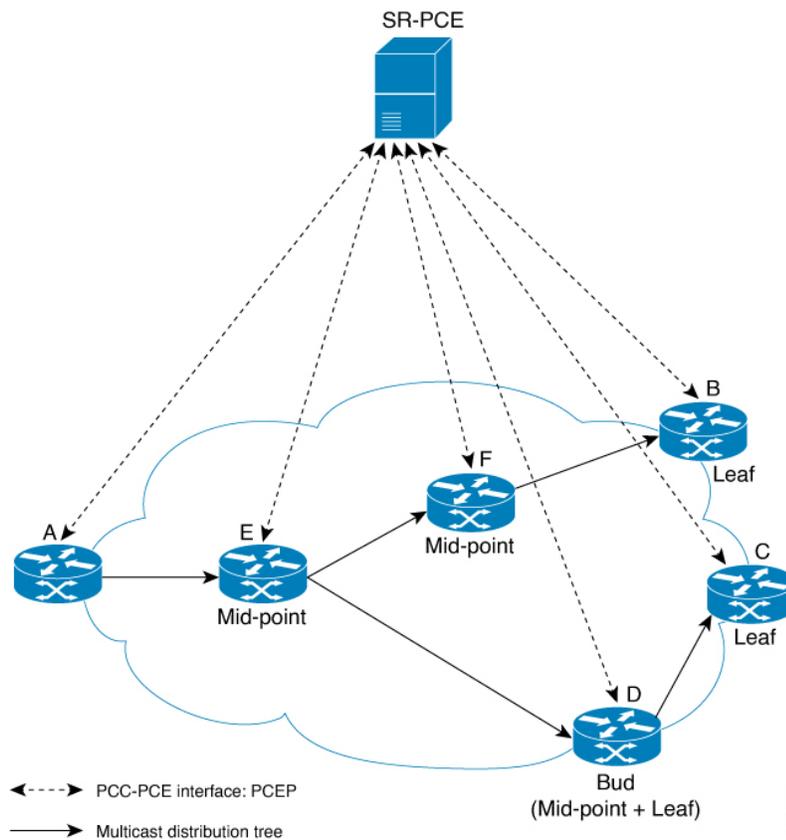
Behaviors and Limitations

PCE redundancy is not supported for Tree-SID. Tree-SID can only be controlled by a single PCE. Configure only one PCE on each PCC in the Tree-SID path.

Bud Node Support

In a multicast distribution tree, a Bud node is a node that acts as a leaf (egress) node as well as a mid-point (transit) node toward the downstream sub-tree.

In the below multicast distribution tree topology with Root node {A} and Leaf nodes set {B, C, D}, node D is a Bud node. Similarly, if node E is later added to the Leaf set, it would also become a Bud node.



The tree computation algorithm on SR-PCE has been enhanced to detect a Bud node based on knowledge of the Leaf set, and to handle Leaf/Transit node transitions to Bud node. The role of the Bud node is also explicitly signaled in PCEP.

Configure Segment Routing Tree-SID

To configure Segment Routing Tree-SID for Point-to-Multipoint (P2MP) SR policies, complete the following configurations:

1. Configure Path Computation Element Protocol (PCEP) Path Computation Client (PCC) on all nodes involved in the Tree-SID path (root, mid-point, leaf)
2. Configure Affinity Maps on the SR-PCE
3. Configure P2MP SR Policy on SR-PCE
4. Configure Multicast on the Root and Leaf Nodes

Configure PCEP PCC on All Nodes in Tree-SID Path

Configure all nodes involved in the Tree-SID path (root, mid-point, leaf) as PCEP PCC. For detailed PCEP PCC configuration information, see [Configure the Head-End Router as PCEP PCC, on page 126](#).

Configure Affinity Maps on the SR-PCE

Use the **affinity bit-map** *COLOR bit-position* command in PCE SR-TE sub-mode to define affinity maps. The bit-position range is from 0 to 255.

```
Router# configure
Router(config)# pce
Router(config-pce)# segment-routing traffic-eng
Router(config-pce-sr-te)# affinity bit-map RED 23
Router(config-pce-sr-te)# affinity bit-map BLUE 24
Router(config-pce-sr-te)# affinity bit-map CROSS 25
Router(config-pce-sr-te)#
```

Configure P2MP SR Policy on SR-PCE

Configure the end-point name and addresses, Tree-SID label, and constraints for the P2MP policy.

Use the **endpoint-set** *NAME* command in SR-PCE P2MP sub-mode to enter the name of the end-point set and to define the set of end-point addresses.

```
Router(config-pce-sr-te)# p2mp
Router(config-pce-sr-te-p2mp)# endpoint-set BAR
Router(config-pce-p2mp-ep-set)# ipv4 10.1.1.2
Router(config-pce-p2mp-ep-set)# ipv4 10.1.1.3
Router(config-pce-p2mp-ep-set)# ipv4 10.1.1.4
Router(config-pce-p2mp-ep-set)# exit
Router(config-pce-sr-te-p2mp)#
```

Use the **policy** *policy* command to configure the P2MP policy name and enter P2MP Policy sub-mode. Configure the source address, endpoint-set color, Tree-SID label, affinity constraints, and metric type.

```
Router(config-pce-sr-te-p2mp)# policy FOO
Router(config-pce-p2mp-policy)# source ipv4 10.1.1.6
Router(config-pce-p2mp-policy)# color 10 endpoint-set BAR
Router(config-pce-p2mp-policy)# treesid mpls 15200
Router(config-pce-p2mp-policy)# candidate-paths
Router(config-pce-p2mp-policy-path)# constraints
Router(config-pce-p2mp-path-const)# affinity
Router(config-pce-p2mp-path-affinity)# exclude BLUE
Router(config-pce-p2mp-path-affinity)# exit
Router(config-pce-p2mp-path-const)# exit
Router(config-pce-p2mp-policy-path)# preference 100
Router(config-pce-p2mp-policy-path-preference)# dynamic
Router(config-pce-p2mp-path-info)# metric type te
Router(config-pce-p2mp-path-info)# root
Router(config)#
```

Configure Multicast on the Root and Leaf Nodes

On the root node of the SR P2MP segment, use the **router pim** command to enter Protocol Independent Multicast (PIM) configuration mode to statically steer multicast flows into an SR P2MP policy.



Note Enter this configuration only on an SR P2MP segment. Multicast traffic cannot be steered into a P2P policy.

```
Router(config)# router pim
Router(config-pim)# vrf name
Router(config-pim-name)# address-family ipv4
Router(config-pim-name-ipv4)# sr-p2mp-policy FOO
```

```
Router(config-pim-name-ipv4-srp2mp) # static-group 235.1.1.5 10.1.1.6
Router(config-pim-name-ipv4-srp2mp) # root
Router(config) #
```

On the root and leaf nodes of the SR P2MP tree, use the **mdt static segment-routing** command to configure the multicast distribution tree (MDT) core as Tree-SID from the multicast VRF configuration submode.

```
Router(config) # multicast-routing
Router(config-mcast) # vrf TEST
Router(config-mcast-TEST) # address-family ipv4
Router(config-mcast-TEST-ipv4) # mdt static segment-routing
```

On the leaf nodes of an SR P2MP segment, use the **static sr-policy p2mp-policy** command to configure the static SR P2MP Policy from the multicast VRF configuration submode to statically decapsulate multicast flows.

```
Router(config) # multicast-routing
Router(config-mcast) # vrf TEST
Router(config-mcast-TEST) # address-family ipv4
Router(config-mcast-TEST-ipv4) # static sr-policy FOO
```

Running Config

The following example shows how to configure the end point addresses and P2MP SR policy with affinity constraints on SR-PCE.

```
pce
  segment-routing
  traffic-eng
  affinity bit-map
  RED 23
  BLUE 24
  CROSS 25
  !
  p2mp
  endpoint-set BAR
  ipv4 10.1.1.2
  ipv4 10.1.1.3
  ipv4 10.1.1.4
  !
  policy FOO
  source ipv4 10.1.1.6
  color 10 endpoint-set BAR
  treesid mpls 15200
  candidate-paths
  preference 100
  dynamic
  metric
  type te
  !
  !
  !
  constraints
  affinity
  exclude
  BLUE
  !
  !
```

```

!
!
!
!
!
!
!

```

The following example shows how to statically decapsulate multicast flows on the leaf nodes.

```

multicast-routing
vrf TEST
  address-family ipv4
    static sr-policy FOO
  !
!
!

```

The following example shows to configure the multicast distribution tree (MDT) core as Tree-SID on the root and leaf nodes.

```

multicast-routing
vrf TEST
  address-family ipv4
    mdt static segment-routing
  !
!
!

```

The following example shows how to steer traffic to the SR P2MP policy on the root node.

```

router pim
vrf TEST
  address-family ipv4
    sr-p2mp-policy FOO
    static-group 232.1.1.5 10.1.1.6
  !
!
!
!

```



CHAPTER 10

Enabling Segment Routing Flexible Algorithm

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

The SR architecture associates prefix-SIDs to an algorithm which defines how the path is computed. Flexible Algorithm allows for user-defined algorithms where the IGP computes paths based on a user-defined combination of metric type and constraint.

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

- [Prerequisites for Flexible Algorithm, on page 159](#)
- [Building Blocks of Segment Routing Flexible Algorithm, on page 159](#)
- [Configuring Flexible Algorithm, on page 162](#)
- [Example: Configuring IS-IS Flexible Algorithm, on page 163](#)
- [Example: Configuring OSPF Flexible Algorithm, on page 164](#)
- [Example: Traffic Steering to Flexible Algorithm Paths, on page 164](#)

Prerequisites for Flexible Algorithm

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

Building Blocks of Segment Routing Flexible Algorithm

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

Flexible Algorithm Definition

Many possible constraints may be used to compute a path over a network. Some networks are deployed with multiple planes. A simple form of constraint may be to use a particular plane. A more sophisticated form of constraint can include some extended metric, like delay, as described in [RFC7810]. Even more advanced case could be to restrict the path and avoid links with certain affinities. Combinations of these are also possible. To provide a maximum flexibility, the mapping between the algorithm value and its meaning can be defined

by the user. When all the routers in the domain have the common understanding what the particular algorithm value represents, the computation for such algorithm is consistent and the traffic is not subject to looping. Here, since the meaning of the algorithm is not defined by any standard, but is defined by the user, it is called a Flexible Algorithm.

Flexible Algorithm Membership

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

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

Flexible Algorithm Definition Advertisement

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

Definition of Flexible Algorithm includes:

- Metric type
- Affinity constraints

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 [#unique_114 unique_114_Connect_42_section_khh_5k1_hwb](#).

- 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 219](#).

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 207](#).

You can disable TI-LFA for Flexible Algorithm using the following commands:

```
router isis instance flex-algo algo fast-reroute disable
router ospf process flex-algo algo fast-reroute disable
```

Installation of Forwarding Entries for Flexible Algorithm Paths

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

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

Flexible Algorithm Prefix-SID Redistribution

Prefix redistribution from IS-IS to another IS-IS instance or protocol was limited to SR algorithm 0 (regular SPF) prefix SIDs; SR algorithm 1 (Strict SPF) and SR algorithms 128-255 (Flexible Algorithm) prefix SIDs were not redistributed along with the prefix. The Segment Routing IS-IS Flexible Algorithm Prefix SID Redistribution feature allows redistribution of strict and flexible algorithms prefix SIDs from IS-IS to another IS-IS instance or protocols. This feature is enabled automatically when you configure redistribution of IS-IS Routes with strict or flexible algorithm SIDs.

Configuring Flexible Algorithm

The following IS-IS and OSPF configuration sub-mode is used to configure Flexible Algorithm:

```
router isis instance flex-algo algo
router ospf process flex-algo algo
algo—value from 128 to 255
```

Configuring Flexible Algorithm Definitions

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

- IS-IS

```
metric-type delay
```



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

OSPF

```
metric-type {delay | te-metric}
```



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

- **affinity** {include-any | include-all | exclude-any} name1, name2, ...

name—name of the affinity map

- **priority** priority value

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

The following command is used to 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: Configuring OSPF Flexible Algorithm

```

router ospf 1
  flex-algo 130
  priority 200
  affinity exclude-any
    red
    blue
  !
  metric-type delay
  !
  flex-algo 140
  affinity include-all
    green
  !
  affinity include-any
    red
  !
  !

  interface Loopback0
    prefix-sid index 10
    prefix-sid strict-spf index 40
    prefix-sid algorithm 128 absolute 16128
    prefix-sid algorithm 129 index 129
    prefix-sid algorithm 200 index 20
    prefix-sid algorithm 210 index 30
  !
  !

  interface GigabitEthernet0/0/0/0
    flex-algo affinity
      color red
      color blue
    !
    !

  affinity-map
    color red bit-position 10
    color blue bit-position 11
  !

```

Example: Traffic Steering to Flexible Algorithm Paths

BGP Routes on PE – Color Based Steering

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

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

Configuration on router R1:

```

vrf Test
  address-family ipv4 unicast
    import route-target

```

```

    1:150
    !
    export route-policy SET_COLOR_RED_HI_BW
    export route-target
    1:150
    !
    !
    !
    interface Loopback0
    ipv4 address 2.2.2.2 255.255.255.255
    !
    interface Loopback150
    vrf Test
    ipv4 address 2.2.2.222 255.255.255.255
    !
    interface TenGigE0/1/0/3/0
    description exr1 to cxr1
    ipv4 address 10.0.20.2 255.255.255.0
    !
    extcommunity-set opaque color129-red-igp
    129
    end-set
    !
    route-policy PASS
    pass
    end-policy
    !
    route-policy SET_COLOR_RED_HI_BW
    set extcommunity color color129-red-igp
    pass
    end-policy
    !
    router isis 1
    is-type level-2-only
    net 49.0001.0000.0000.0002.00
    log adjacency changes
    affinity-map RED bit-position 28
    flex-algo 128
    priority 228
    !
    address-family ipv4 unicast
    metric-style wide
    advertise link attributes
    router-id 2.2.2.2
    segment-routing mpls
    !
    interface Loopback0
    address-family ipv4 unicast
    prefix-sid index 2
    prefix-sid algorithm 128 index 282
    !
    !
    !
    interface TenGigE0/1/0/3/0
    point-to-point
    address-family ipv4 unicast
    !
    !
    !
    router bgp 65000
    bgp router-id 2.2.2.2
    address-family ipv4 unicast
    !
    address-family vpnv4 unicast
    retain route-target all

```



```
    pass
end-policy
!
router isis 1
is-type level-2-only
net 49.0001.0000.0000.0004.00
log adjacency changes
affinity-map RED bit-position 28
affinity-map BLUE bit-position 29
affinity-map GREEN bit-position 30
flex-algo 128
    priority 228
!
flex-algo 129
    priority 229
!
flex-algo 130
    priority 230
!
address-family ipv4 unicast
    metric-style wide
    advertise link attributes
    router-id 4.4.4.4
    segment-routing mpls
!
interface Loopback0
    address-family ipv4 unicast
        prefix-sid index 4
        prefix-sid algorithm 128 index 284
        prefix-sid algorithm 129 index 294
        prefix-sid algorithm 130 index 304
!
!
interface GigabitEthernet0/0/0/0
    point-to-point
    address-family ipv4 unicast
!
!
interface TenGigE0/1/0/1
    point-to-point
    address-family ipv4 unicast
!
!
router bgp 65000
bgp router-id 4.4.4.4
address-family ipv4 unicast
!
address-family vpnv4 unicast
!
neighbor-group RR-services-group
    remote-as 65000
    update-source Loopback0
    address-family ipv4 unicast
!
    address-family vpnv4 unicast
!
!
neighbor 10.1.1.1
    use neighbor-group RR-services-group
!
neighbor 2.2.2.2
    use neighbor-group RR-services-group
!
vrf Test
```

```
rd auto
address-family ipv4 unicast
  redistribute connected
!
neighbor 25.1.1.2
  remote-as 4
  address-family ipv4 unicast
    route-policy PASS in
    route-policy PASS out
!
!
!
segment-routing
!
end
```



CHAPTER 11

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 169](#)
- [Configure SR-PCE, on page 170](#)
- [PCE-Initiated SR Policies, on page 174](#)
- [ACL Support for PCEP Connection, on page 175](#)
- [Anycast SID-Aware Path Computation, on page 176](#)
- [PCE Support for MPLS-TE LSPs, on page 179](#)
- [Configuring the North-Bound API on SR-PCE, on page 182](#)

About SR-PCE

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

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

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

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

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

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

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# configure	Enters global configuration mode.
Step 2	pce Example: RP/0/RP0/CPU0:router(config)# pce	Enables PCE and enters PCE configuration mode.
Step 3	address ipv4 <i>address</i> Example: RP/0/RP0/CPU0:router(config-pce)# address ipv4	Configures a PCE IPv4 address.

	Command or Action	Purpose
	192.168.0.1	
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.
Step 10	minimum-peer-keepalive <i>time</i> Example: RP/0/RP0/CPU0:router(config-pce-timers) # minimum-peer-keepalive 30	Configures the minimum acceptable keep-alive timer that the remote peer may propose in the PCEP OPEN message during session establishment. The default time is 20 seconds.

	Command or Action	Purpose
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: <pre>RP/0/RP0/CPU0:router(config-pce-disjoint)# group-id 1 type node sub-id 1</pre>	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): <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.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • 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	strict Example: <pre>RP/0/RP0/CPU0:router(config-pce-disjoint)# strict</pre>	(Optional) Prevents the automatic fallback behavior of the preferred level of disjointness. If a pair of paths that meet the requested disjointness level cannot be found, the disjoint calculation terminates and no new path is provided. The existing path is not modified.
Step 4	lsp {1 2} pcc ipv4 address lsp-name lsp_name [shortest-path] Example: <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>	Adds LSPs to the disjoint group. The shortest-path keyword forces one of the disjoint paths to follow the shortest path from the source to the destination. This option can only be applied to the the first LSP specified.

Global Maximum-delay Constraint

This feature allows a PCE to compare the cumulative latency of a computed path against a global maximum-delay constraint value. If the latency of the computed path exceeds this global constraint, the path is not considered valid. This ensures that all latency-based paths computed by the PCE and signaled to the PCCs in the network do not exceed this maximum-delay constraint.

```
pce
  constraints
    bounds
      cumulative
        type
          latency <1-4294967295> Bound metric value in microseconds
```

Configuration

To configure a PCE for specifying maximum cumulative latency metric, you must complete the following configurations:

```
RP/0/RSP0/CPU0:ios(config)# pce
RP/0/RSP0/CPU0:ios(config-pce)# constraints
RP/0/RSP0/CPU0:ios(config-pce-constr)# bounds
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds)# cumulative
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds-type)# type latency 1000000
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds-type)#
```

Verification

Verify using the **show** command:

```
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds-type)# show
Wed Oct 12 22:18:22.962 UTC
pce
  constraints
  bounds
  cumulative
  type latency 1000000
  !
  !
  !
  !
```

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 83](#).

The PCE deploys the SR-TE policy using PCC-PCE communication protocol (PCEP).

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

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

Configuration Example: PCE-Initiated SR Policy with Explicit SID List

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

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



Note When configuring an explicit path using IP addresses of intermediate links, the SR-TE process prefers the protected Adj-SID of the link, if one is available.

3. Create the policy.

```

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

/* Create the SR-TE segment lists */
Router(config-pce)# segment-routing
Router(config-pce-sr)# traffic-eng
Router(config-pce-sr-te)# segment-list name addr2a
Router(config-pce-sr-te-sl)# index 10 address ipv4 10.1.1.2
Router(config-pce-sr-te-sl)# index 20 address ipv4 10.2.3.2
Router(config-pce-sr-te-sl)# index 30 address ipv4 10.1.1.4
Router(config-pce-sr-te-sl)# exit

/* Create the SR-TE policy */
Router(config-pce-sr-te)# peer ipv4 10.1.1.1
Router(config-pce-sr-te)# policy P1
Router(config-pce-sr-te-policy)# color 2 end-point ipv4 2.2.2.2
Router(config-pce-sr-te-policy)# candidate-paths
Router(config-pce-sr-te-policy-path)# preference 50
Router(config-pce-sr-te-policy-path-preference)# explicit segment-list addr2a
Router(config-pce-sr-te-pp-info)# commit
Router(config-pce-sr-te-pp-info)# end
Router(config)#

```

Running Config

```

pce
 segment-routing
  traffic-eng
  segment-list name addr2a
  index 10 address ipv4 10.1.1.2
  index 20 address ipv4 10.2.3.2
  index 30 address ipv4 10.1.1.4
  !
 peer ipv4 10.1.1.1
  policy P1
  color 2 end-point ipv4 2.2.2.2
  candidate-paths
  preference 50
  explicit segment-list addr2a
  !
 !

```

ACL Support for PCEP Connection

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

The ACL Support for PCEP Connection feature provides a way to protect a PCE server using an Access Control List (ACL) to restrict IPv4 PCC peers at the time the TCP connection is created based on the source address of a client. When a client initiates the TCP connection, the ACL is referenced, and the client source

address is compared. The ACL can either permit or deny the address and the TCP connection will proceed or not.

Refer to the Implementing Access Lists and Prefix Lists chapter in the *IP Addresses and Services Configuration Guide for Cisco NCS 5500 Series Routers* for detailed ACL configuration information.

To apply an ACL to the PCE, use the **pce peer-filter ipv4 access-list** *acl_name* command.

The following example shows how to configure an ACL and apply it to the PCE:

```
pce
 address ipv4 10.1.1.5
 peer-filter ipv4 access-list sample-peer-filter
 !
ipv4 access-list sample-peer-filter
 10 permit ipv4 host 10.1.1.6 any
 20 permit ipv4 host 10.1.1.7 any
 30 deny ipv4 any any
 !
```

Anycast SID-Aware Path Computation

This feature allows the SR-TE head-end or SR-PCE to compute a path that is encoded using Anycast prefix SIDs of nodes along the path.

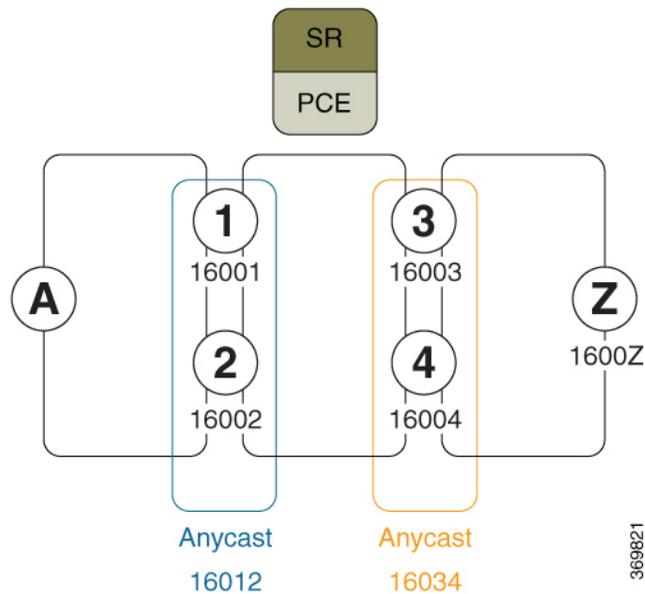
An Anycast SID is a type of prefix SID that identifies a set of nodes and is configured with n-flag clear. The set of nodes (Anycast group) is configured to advertise a shared prefix address and prefix SID. Anycast routing enables the steering of traffic toward multiple advertising nodes, providing load-balancing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



Note For information on configuring Anycast SID, see [Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 43](#) and [Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 63](#).

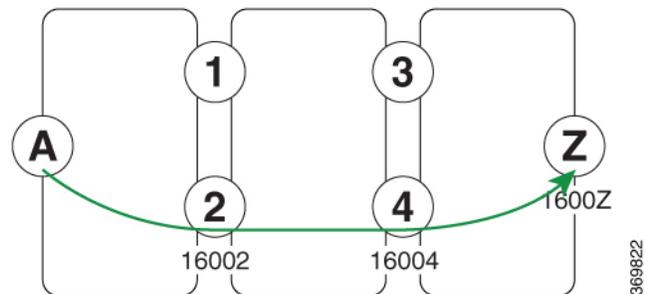
This example shows how Anycast SIDs are inserted into a computed SID list.

The following figure shows 3 isolated IGP domains without redistribution and without BGP 3107. Each Area Border Router (ABR) 1 through 4 is configured with a node SID. ABRs 1 and 2 share Anycast SID 16012 and ABRs 3 and 4 share Anycast SID 16034.



Consider the case where nodes A and Z are provider edge (PE) routers in the same VPN. Node A receives a VPN route with BGP next-hop to node Z. Node A resolves the SR path to node Z based on ODN behaviors with delegation of path computation to SR-PCE.

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



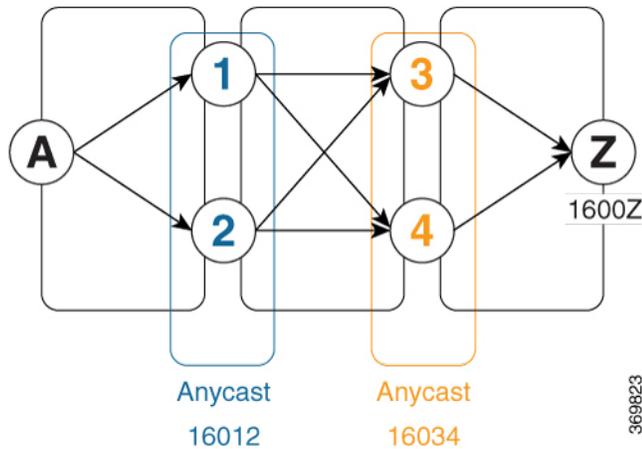
Assume that the computed path from node A to node Z traverses node 2 and node 4. This translates to SID list {16002, 16004, 1600Z} when node SIDs are leveraged to encode the path.

When an Anycast SID-aware path is requested, the path computation algorithm performs the following:

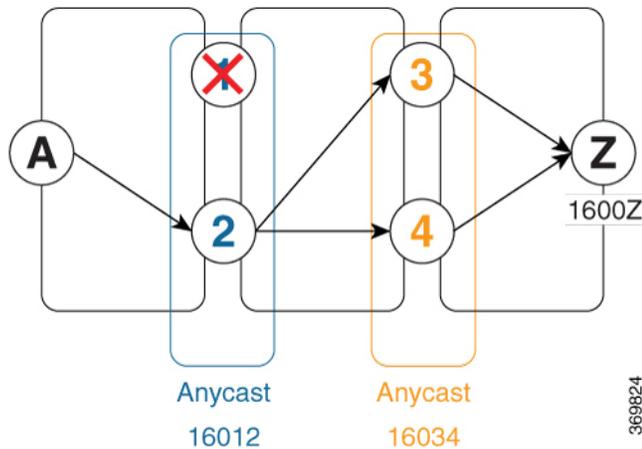
- **Path Computation**—Computes the path according to optimization objectives and constraints
- **Path Encoding**—Encodes the path in a SID list leveraging node-SIDs and adj-SIDs as applicable
- **Anycast SID Replacement**—Reiterates the original SID list by replacing node SIDs with Anycast SIDs present on the nodes along the computed path.
- **Optimality Validation**—The new paths are validated against the original optimization objectives and constraints (maintain same cumulative metric as original SID list and do not violate path constraints).
- **Anycast SID Promotion**—If the optimality validation is successful, then the Anycast-encoded SID list is signaled and instantiated in the forwarding.

The following figure depicts cumulative metrics between nodes in the network.

Under these conditions, the optimality check is met, and therefore, the Anycast-encoded SID list from node A to node Z is {16012,16034,1600Z}.



The Anycast SID aware path computation also provides resiliency. For example, if one of the ABRs (in this case, ABR 1) becomes unavailable or unreachable, the path from node A to node Z {16012,16034,1600Z} will still be valid and usable.



Configuration Examples

1. Configure Prefix SIDs on the ABR nodes.
 - a. Configure each node with a node SID.
 - b. Configure each group of nodes with a shared Anycast SID.

See [Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface, on page 43](#) and [Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface, on page 63](#).

2. Configure SR policies to include Anycast SIDs for path computation using the **anycast-sid-inclusion** command.

This example shows how to configure a local SR policy to include Anycast SIDs for PCC-initiated path computation at the head-end router:

```

Router(config)# segment-routing traffic-eng
Router(config-sr-te)# policy FOO
Router(config-sr-te-policy)# color 10 end-point ipv4 10.1.1.10
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# dynamic
Router(config-sr-te-pp-info)# anycast-sid-inclusion

```

Running Configuration

Use the **anycast-sid-inclusion** command to request Anycast SID-aware path computation for the following SR policy types:

- Local SR policy with PCC-initiated path computation at the head-end router:

```

segment-routing
 traffic-eng
  policy FOO
    color 10 end-point ipv4 10.1.1.10
    candidate-paths
      preference 100
      dynamic
      anycast-sid-inclusion

```

- Local SR policy with PCC-initiated/PCE-delegated path computation at the SR-PCE:

```

segment-routing
 traffic-eng
  policy BAR
    color 20 end-point ipv4 10.1.1.20
    candidate-paths
      preference 100
      dynamic
      pcep
      anycast-sid-inclusion

```

- On-demand SR policies with a locally computed dynamic path at the head-end, or centrally computed dynamic path at the SR-PCE:

```

segment-routing
 traffic-eng
  on-demand color 10
  dynamic
  anycast-sid-inclusion

```

- On-demand SR policies with centrally computed dynamic path at the SR-PCE:

```

segment-routing
 traffic-eng
  on-demand color 20
  dynamic
  pcep
  anycast-sid-inclusion

```

PCE Support for MPLS-TE LSPs

This feature allows Cisco's SR-PCE to act as a Path Computation Element (PCE) for MPLS Traffic Engineering Label Switched Paths (MPLS-TE LSPs).



Note For more information about MPLS-TE, refer to the "Implementing MPLS Traffic Engineering" chapter in the *MPLS Configuration Guide for Cisco NCS 5500 Series Routers*.

The supported functionality is summarized below:

- PCE type: Active Stateful PCE
- MPLS-TE LSP initiation methods:
 - PCE Initiated—An active stateful PCE initiates an LSP and maintains the responsibility of updating the LSP.
 - PCC Initiated—A PCC initiates the LSP and may delegate the control later to the Active stateful PCE.
- MPLS-TE LSP metric—Metric optimized by the path computation algorithm:
 - IGP metric
 - TE metric
 - Latency metric
- MPLS-TE LSP constraints—TE LSP attributes to be taken into account by the PCE during path computation:
 - Resource Affinities
 - Path Disjointness
- MPLS-TE LSP parameters:
 - Setup priority—The priority of the TE LSP with respect to taking resources
 - Hold priority—The priority of the TE LSP with respect to holding resources
 - FRR L flag—The "Local Protection Desired" bit. Can be set from an application instantiating an MPLS-TE LSP via SR-PCE. SR-PCE passes this flag to the PCC, and the PCC will enable FRR for that LSP.
 - Signaled Bandwidth—This value can be set from an application instantiating an MPLS-TE LSP via SR-PCE. SR-PCE passes this value to the PCC.
 - Binding SID—A segment identifier (SID) that a headend binds to an MPLS-TE LSP. When the headend receives a packet with active segment (top MPLS label) matching the BSID of a local MPLS-TE LSP, the headend steers the packet into the associated MPLS-TE LSP.

Cisco Crosswork Optimization Engine is an application that leverages the SR-PCE in order to visualize and instantiate MPLS-TE LSPs. For more information, refer to the [Visualize SR Policies and RSVP-TE Tunnels](#) chapter in the [Cisco Crosswork Optimization Engine 1.2.1 User Guide](#).



Note No extra configuration is required to enable MPLS-TE support at SR-PCE.

Example: Configuring a PCEP Session (Stateful Mode) on MPLS-TE PCC

The following example shows the configuration for an MPLS-TE PCC to establish a PCEP session with a PCE (IPv4 address 10.1.1.100).



Note MPLS-TE PCC must operate in the stateful PCEP mode when connecting to SR-PCE.

The **instantiation** keyword enables the PCC to support MPLS-TE LSP instantiation by PCE (PCE-initiated).

The **report** keyword enables the PCC to report all the MPLS-TE LSPs configured on that node.



Note PCE-initiated LSPs are automatically reported to all configured PCEs.

The **autoroute-announce** keyword enables autoroute-announce globally for all PCE-initiated LSPs on the PCC.

The **redundancy pcc-centric** keywords enable PCC-centric high-availability model for PCE-initiated LSPs. The PCC-centric model changes the default PCC delegation behavior to the following:

- After LSP creation, LSP is automatically delegated to the PCE that computed it.
- If this PCE is disconnected, then the LSP is redelegated to another PCE.
- If the original PCE is reconnected, then the delegation fallback timer is started. When the timer expires, the LSP is redelegated back to the original PCE, even if it has worse preference than the current PCE.

```
mpls traffic-eng
 pce
  peer ipv4 10.1.1.100
  !
  stateful-client
  instantiation
  report
  autoroute-announce
  redundancy pcc-centric
  !
  !
  !
end
```

Example: Configuring Multiple PCEP Sessions from a PCC Acting as MPLS-TE and SR-TE Headend Toward a Common PCE

The following example shows the configuration for a PCC (IPv4 addresses 10.1.1.1 and 10.1.1.2) to establish two PCEP sessions with a common PCE (IPv4 address 10.1.1.100). One session is configured under MPLS-TE, and the other under SR-TE.



Note The two PCEP sessions must use a different source address on the PCC when connecting to the same PCE.

For more information regarding PCEP configuration at SR-TE PCC, see the *Configure the Head-End Router as PCEP PCC* topic.

```

mpls traffic-eng
  pce
    peer source ipv4 10.1.1.1
    peer ipv4 10.1.1.100
    !
  !
end

segment-routing
  traffic-eng
    pcc
      source-address ipv4 10.1.1.2
      pce address ipv4 10.1.1.100
      !
    !
  !
end

```

Configuring the North-Bound API on SR-PCE

The SR-PCE provides a north-bound HTTP-based API to allow communication between SR-PCE and external clients and applications.

Over this API, an external application can leverage the SR-PCE for topology discovery, SR policy discovery, and SR policy instantiation.

The Cisco Crosswork Optimization Engine is an application that leverages the SR-PCE. For more information, refer to the [Cisco Crosswork Optimization Engine User Guides](#).

Use the following commands under PCE configuration mode to configure the API to allow communication between SR-PCE and external clients or applications.



Note The API server is enabled by default when SR-PCE is configured.

Command	Description
rest authentication basic	(Optional) Specify basic (plaintext) authentication. By default, authentication is disabled.
rest username <i>password</i> {clear encrypted} <i>password</i>	Add credentials when connecting to API. Note This command is used only if authentication is configured.

Command	Description
<code>rest sibling ipv4 address</code>	<p>Opens a synchronization channel to another PCE in the same high availability (HA) pair.</p> <p>Note For more information regarding SR-PCE HA pairs, refer to the Multiple Cisco SR-PCE HA Pairs chapter of the Cisco Crosswork Optimization Engine 1.2.1 User Guide.</p>

Example: Configuring API on SR-PCE

```
pce
address ipv4 10.1.1.100
rest
  user admin
  password encrypted 1304131F0202
  !
  authentication basic
  sibling ipv4 10.1.1.200
  !
  !
end
```

The following example shows the current active connections:

```
RP/0/0/CPU0:pce1# show tcp brief | i 8080
Thu Aug 6 00:40:15.408 PDT
0xe9806fb8 0x60000000 0 0 :::8080 :::0 LISTEN
0xe94023b8 0x60000000 0 0 10.1.1.100:50487 10.1.1.200:8080 ESTAB
0xeb20bb40 0x60000000 0 0 10.1.1.100:8080 10.1.1.200:44401 ESTAB
0xe98031a0 0x60000000 0 0 0.0.0.0:8080 0.0.0.0:0 LISTEN
```

The first and fourth entries show the API server listening for IPv4 and IPv6 connections.

The second and third entries show the established sibling connection between PCE1 (10.1.1.100) and PCE2 (10.1.1.200).



CHAPTER 12

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 6: Performance Measurement Functionalities

Functionality	Details
Profiles	You can configure different 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.

- [Measurement Modes](#), on page 186
- [Usage Guidelines and Limitations](#), on page 187
- [Link Delay Measurement](#), on page 188
- [SR Policy End-to-End Delay Measurement](#), on page 199
- [SR Policy Liveness Monitoring - Hardware Offloading](#), on page 206

Measurement Modes

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

Table 7: Measurement Mode Requirements

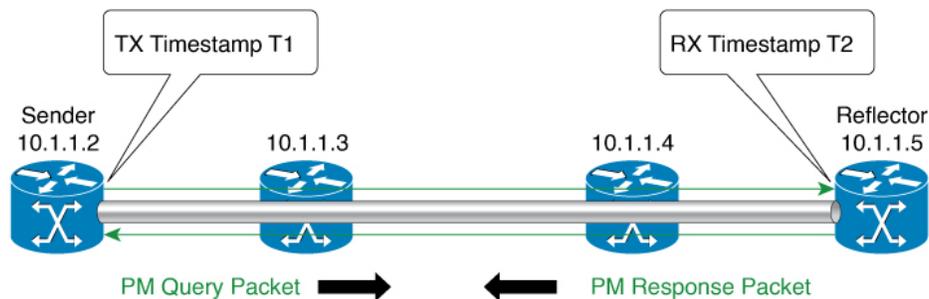
Measurement Mode	Sender: PTP-Capable HW and HW Timestamping	Reflector: PTP-Capable HW and HW Timestamping	PTP Clock Synchronization between Sender and Reflector
One-way	Required	Required	Required
Two-way	Required	Required	Not Required

One-Way

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 6: 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.

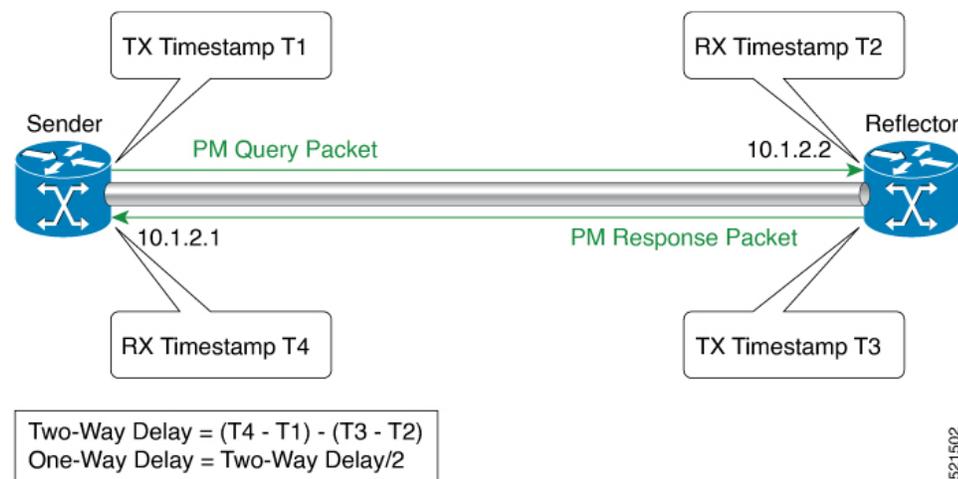
Two-Way

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 measurements in two-way mode are calculated as follows:

- Two-Way Delay = $(T4 - T1) - (T3 - T2)$
- One-Way Delay = Two-Way Delay/2

Figure 7: Two-Way



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

1. The local-end router sends PM query packets periodically to the remote side once the egress line card on the router applies timestamps on packets.
2. Ingress line card on the remote-end router applies time-stamps on packets as soon as they are received.
3. The remote-end router sends the PM packets containing time-stamps back to the local-end router. The remote-end router time-stamps the packet just before sending it for two-way measurement.
4. The local-end router time-stamps the packet as soon as the packet is received for two-way measurement.
5. One-way delay and optionally two-way delay is measured using the time-stamp values in the PM packet.

Usage Guidelines and Limitations

The following usage guidelines and limitations apply:

- SR PM is supported on hardware that supports Precision Time Protocol (PTP). This requirement applies to both one-way and two-way delay measurement.

See the [Configuring Precision Time Protocol](#) chapter in the *System Management Configuration Guide for Cisco NCS 5500 Series Routers* for Restrictions for PTP and the Timing Hardware Support Matrix.

Link Delay Measurement

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 186](#) 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)# next-hop ipv4 10.10.10.2 // Optional IPv4 or IPv6
next-hop address
RP/0/0/CPU0:router(config-pm-intf)# delay-measurement
RP/0/0/CPU0:router(config-pm-intf-dm)# exit
```

The source and destination IP addresses used in the OAM packet are determined by the IP address present on the interface where the delay-measurement operation is enabled and the setting of the optional **next-hop** address.

When the **next-hop** address is not specified, the following rules apply to determine the source and destination IP addresses used in the OAM packet:

- If an IPv4 address is configured under the interface, then:
 - OAM packet source IP address = Interface's IPv4 address
 - OAM packet destination IP address = 127.0.0.0
- Else, if an IPv6 global address is configured under the interface, then:
 - OAM packet source IP address = Interface's IPv6 global address
 - OAM packet destination IP address = 0::ff:127.0.0.0

When the **next-hop** {**ipv4** | **ipv6**} address is configured, the following rules apply to determine the source and destination IP addresses used in the OAM packet:

- If a next-hop IPv4 address is configured, then:
 - OAM packet source IP address = Interface's IPv4 address
 - OAM packet destination IP address = Configured next-hop IPv4 address



Note If there is no IPv4 address configured under the interface, then the delay-measurement probe does not send OAM packets.

- If a next-hop IPv6 address is configured, then:
 - OAM packet source IP address = Interface's IPv6 global address
 - OAM packet destination IP address = Configured next-hop IPv6 address



Note If there is no IPv6 global address configured under the interface, then the delay-measurement probe does not send OAM packets.

This example shows how to enable PM for link delay over an interface with IPv4 address configured:

```
interface TenGigE0/0/0/0
  ipv4 address 10.10.10.1 255.255.255.0

performance-measurement
  interface TenGigE0/0/0/0
    delay-measurement
```

This example shows how to enable PM for link delay over an interface IPv6 address configured:

```

interface TenGigE0/0/0/0
  ipv6 address 10:10:10::1/64

performance-measurement
  interface TenGigE0/0/0/0
    delay-measurement

```

This example shows how to enable PM for link delay over an interface with a specified next-hop IPv4 address:

```

interface TenGigE0/0/0/0
  ipv4 address 10.10.10.1 255.255.255.0

performance-measurement
  interface TenGigE0/0/0/0
    next-hop ipv4 10.10.10.2
    delay-measurement

```

This example shows how to enable PM for link delay over an interface with a specified next-hop IPv6 address:

```

interface TenGigE0/0/0/0
  ipv6 address 10:10:10::1/64

performance-measurement
  interface TenGigE0/0/0/0
    next-hop ipv6 10:10:10::2
    delay-measurement

```

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 Sent: 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 name 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 sr-policy name profile3
```

```
-----
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 profile3
  !

```

Verification

```

Router# show performance-measurement profile named-profile liveness sr-policy name profile3

```

```

-----
0/RSP0/CPU0
-----
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

```

SR Policy End-to-End Delay Measurement

The PM for SR Policy uses the IP/UDP packet format defined in RFC 5357 (TWAMP-Light) for probes. Two-Way Active Measurement Protocol (TWAMP) adds two-way or round-trip measurement capabilities. TWAMP employs time stamps applied at the echo destination (reflector) to enable greater accuracy. In the case of TWAMP Light, the Session-Reflector doesn't necessarily know about the session state. The Session-Reflector simply copies the Sequence Number of the received packet to the Sequence Number field of the reflected packet. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

The extended TE link delay metric (minimum-delay value) can be used to compute paths for SR policies as an optimization metric or as an accumulated delay bound.

There is a need to monitor the end-to-end delay experienced by the traffic sent over an SR policy to ensure that the delay does not exceed the requested "upper-bound" and violate SLAs. You can verify the end-to-end delay values before activating the candidate-path or the segment lists of the SR policy in forwarding table, or to deactivate the active candidate-path or the segment lists of the SR policy in forwarding table.



Note The end-to-end delay value of an SR policy will be different than the path computation result (for example, the sum of TE link delay metrics) due to several factors, such as queuing delay within the routers.

Restrictions and Usage Guidelines for PM for SR Policy Delay

Hardware clocks must be synchronized between the querier and the responder nodes of the link using PTP for one-way delay measurement.

Configuring Performance Measurement Parameters

This example shows how to configure performance-measurement parameters for SR policy delay as a global default profile. The default values for the different parameters in the PM for SR policy delay is given as follows:

- **probe**: The default mode for probe is one-way delay measurement. See [Measurement Modes, on page 186](#) for more information.
- **burst interval**: Interval for sending probe packet. The default value is 3000 milliseconds and the range is from 30 to 15000 milliseconds.
- **computation interval**: Interval for metric computation. Default is 30 seconds; range is 1 to 3600 seconds.
- **protocol**:
 - **twamp-light**: SR Policy delay measurement using RFC 5357 with IP/UDP encap. This is the default protocol.
- **tos**: Type of Service
 - **dscp value**: The default value is 48 and the range is from 0 to 63.
 - **traffic-class value**: The default value is 6 and the range is from 0 to 7.
- **advertisement threshold-check**: minimum-delay/maximum-delay - The default value of periodic advertisement threshold-check is maximum-delay.
- **periodic advertisement**: Periodic advertisement is enabled by default.
- **periodic-advertisement interval**: The default value is 120 seconds and the interval range is from 30 to 3600 seconds.
- **periodic-advertisement threshold**: Checks the minimum-delay metric change for threshold crossing for periodic advertisement. The default value is 10 percent and the range is from 0 to 100 percent.
- **periodic-advertisement minimum-change**: The default value is 500 microseconds (usec) and the range is from 0 to 100000 microseconds.
- **accelerated advertisement**: Accelerated advertisement is disabled by default.
- **accelerated-advertisement threshold**: Checks the minimum-delay metric change for threshold crossing for accelerated advertisement. The default value is 20 percent and the range is from 0 to 100 percent.
- **accelerated-advertisement minimum**: The default value is 500 microseconds and the range is from 1 to 100000 microseconds.

```
Router(config)# performance-measurement delay-profile sr-policy
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# burst-interval 60
Router(config-pm-dm-srpolicy-probe)# computation-interval 60
Router(config-pm-dm-srpolicy-probe)# protocol twamp-light
Router(config-pm-dm-srpolicy-probe)# tos dscp 63
Router(config-pm-dm-srpolicy-probe)# exit
```

```

Router(config-pm-dm-srpolicy) # advertisement
Router(config-pm-dm-srpolicy-adv) # periodic
Router(config-pm-dm-srpolicy-adv-per) # interval 60
Router(config-pm-dm-srpolicy-adv-per) # minimum-change 1000
Router(config-pm-dm-srpolicy-adv-per) # threshold 20
Router(config-pm-dm-srpolicy-adv-per) # exit

Router(config-pm-dm-srpolicy-adv) # accelerated
Router(config-pm-dm-srpolicy-adv-acc) # minimum-change 1000
Router(config-pm-dm-srpolicy-adv-acc) # threshold 10
Router(config-pm-dm-srpolicy-adv-acc) # exit

Router(config-pm-dm-srpolicy-adv) # threshold-check minimum-delay
Router(config-pm-dm-srpolicy-adv) # exit
Router(config-pm-dm-srpolicy) #

```

Configure the UDP Destination Port

Configuring the UDP port for TWAMP-Light protocol is optional. By default, PM uses port 862 as the TWAMP-reserved UDP destination port for delay.

The UDP port is configured for each PM measurement probe type (delay, loss, protocol, authentication mode, etc.) on querier and responder nodes. If you configure a different UDP port, the UDP port for each PM measurement probe type must match on the querier and the responder nodes.



Note The same UDP destination port is used for delay measurement for links and SR Policy.

This example shows how to configure the UDP destination port for delay.

```

Router(config) # performance-measurement
Router(config-perf-meas) # protocol twamp-light
Router(config-pm-protocol) # measurement delay unauthenticated
Router(config-pm-proto-mode) # querier-dst-port 12000

```

Enable Performance Measurement for SR Policy

This example shows how to enable PM for SR policy delay for a specific policy.

```

Router(config) # segment-routing traffic-eng
Router(config-sr-te) # policy foo
Router(config-sr-te-policy) # performance-measurement
Router(config-sr-te-policy-perf-meas) # delay-measurement

```

SR Policy Probe IP/UDP ECMP Hashing Configuration

This example shows how to configure SR Policy ECMP IP-hashing mode.

- The destination IPv4 address 127.x.x.x – 127.y.y.y is used in the Probe messages to take advantages of 3-tuple IP hashing (source-address, destination-address, and local router ID) for ECMP paths of SR-MPLS Policy.



Note The destination IPv4 address must be 127/8 range (loopback), otherwise it will be rejected.

- One PM session is always created for the actual endpoint address of the SR Policy.
- You can specify the number of IP addresses to sweep. The range is from 0 (default, no sweeping) to 128.
- Platforms may have a limitation for large label stack size to not check IP address for hashing.

```
Router(config)# performance-measurement delay-profile sr-policy
Router(config-pm-dm-srpolicy)# probe
Router(config-pm-dm-srpolicy-probe)# sweep
Router(config-pm-dm-srpolicy-probe-sweep)# destination ipv4 127.0.0.1 range 28
```

Verification

```
Router# show performance-measurement sr-policy brief
Mon Jan 20 18:48:41.002 PST
```

```
-----
0/0/CPU0
-----
```

Policy Name	LSP ID	Tx/Rx	Avg/Min/Max/Variance
srte_c_10_ep_192.168.0.4	2	6/6	27012/26906/27203/106

```
Router# show performance-measurement sr-policy name srte_c_10_ep_192.168.0.4 detail verbose
Mon Jan 20 18:44:22.400 PST
```

```
-----
0/0/CPU0
-----
```

```
SR Policy name: srte_c_10_ep_192.168.0.4
```

```
Color : 10
Endpoint : 192.168.0.4
Number of candidate-paths : 1
```

```
Candidate-Path:
```

```
Instance : 2
Preference : 100
Protocol-origin : Configured
Discriminator : 100
Source address : 192.168.0.2
Reverse path label : Not configured
Number of segment-lists : 1
```

```
Last advertisement:
```

```
No advertisements have occurred
```

```
Next advertisement:
```

```
Check scheduled at the end of the current probe (roughly every 30 seconds)
Aggregated delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
Rolling average (uSec): 45218
```

```
Last probe:
```

```
Packets Sent: 9, received: 9
Measured delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
```

```
Current Probe:
```

```
Started at Jan 20 2020 18:44:19.170 (3.453 seconds ago)
Packets Sent: 3, received: 3
```

```

Measured delays (uSec): avg: 26588, min: 26558, max: 26630, variance: 30
Next probe scheduled at Jan 20 2020 18:44:34.166 (in 11.543 seconds)
Next burst packet will be sent in 1.543 seconds
Burst packet sent every 5.0 seconds
Liveness Detection: Disabled

Segment-List          : R4
  16004
Number of atomic paths : 3
Last advertisement:
  No advertisements have occurred
Next advertisement:
  Aggregated delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
  Rolling average (uSec): 45218
Last probe:
  Packets Sent: 9, received: 9
  Measured delays (uSec): avg: 45218, min: 26512, max: 82600, variance: 18706
Current probe:
  Packets Sent: 3, received: 3
  Measured delays (uSec): avg: 26588, min: 26558, max: 26630, variance: 30
Liveness Detection: Disabled

Atomic path:
Hops          : 127.0.0.0
Session ID    : 33554434
Last advertisement:
  No advertisements have occurred
Next advertisement:
  Aggregated delays (uSec): avg: 45407, min: 26629, max: 82600, variance: 18778
  Rolling average (uSec): 45407
Last Probe:
  Packets Sent: 3, received: 3
  Measured delays (uSec): avg: 45407, min: 26629, max: 82600, variance: 18778
Current Probe:
  Packets Sent: 1, received: 1
  Measured delays (uSec): avg: 26630, min: 26630, max: 26630, variance: 0
Probe samples:
  Packet Rx Timestamp      Measured Delay (nsec)
  Jan 20 2020 18:44:19.198      26630730
Liveness Detection: Disabled

Atomic path:
Hops          : 127.0.0.1
Session ID    : 33554435
Last advertisement:
  No advertisements have occurred
Next advertisement:
  Aggregated delays (uSec): avg: 45128, min: 26521, max: 81961, variance: 18607
  Rolling average (uSec): 45128
Last Probe:
  Packets Sent: 3, received: 3
  Measured delays (uSec): avg: 45128, min: 26521, max: 81961, variance: 18607
Current Probe:
  Packets Sent: 1, received: 1
  Measured delays (uSec): avg: 26576, min: 26576, max: 26576, variance: 0
Probe samples:
  Packet Rx Timestamp      Measured Delay (nsec)
  Jan 20 2020 18:44:19.198      26576938
Liveness Detection: Disabled

Atomic path:
Hops          : 192.168.0.4
Session ID    : 33554433
Last advertisement:

```

```

No advertisements have occurred
Next advertisement:
  Aggregated delays (uSec): avg: 45119, min: 26512, max: 81956, variance: 18607
  Rolling average (uSec): 45119
Last Probe:
  Packets Sent: 3, received: 3
  Measured delays (uSec): avg: 45119, min: 26512, max: 81956, variance: 18607
Current Probe:
  Packets Sent: 1, received: 1
  Measured delays (uSec): avg: 26558, min: 26558, max: 26558, variance: 0
Probe samples:
  Packet Rx Timestamp      Measured Delay (nsec)
  Jan 20 2020 18:44:19.198      26558375
Liveness Detection: Disabled

```

```

Router# show performance-measurement history probe sr-policy
Mon Jan 20 18:46:55.445 PST

```

```

-----
0/0/CPU0
-----

```

```

SR Policy name: srte_c_10_ep_192.168.0.4
Color           : 10
Endpoint        : 192.168.0.4

```

```

Candidate-Path:
Preference      : 100
Protocol-origin : Configured
Discriminator   : 100

```

```

Delay-Measurement history (uSec):
  Probe Start Timestamp      Pkt (TX/RX)      Average      Min      Max
  Jan 20 2020 18:46:34.174    9/9      26880      26684      27070
  Jan 20 2020 18:46:19.174    9/9      26899      26822      27004
  Jan 20 2020 18:46:04.173    9/9      26813      26571      27164
  Jan 20 2020 18:45:49.172    9/9      26985      26713      27293
  Jan 20 2020 18:45:34.172    9/9      26744      26557      27005
  Jan 20 2020 18:45:19.171    9/9      26740      26435      27093
  Jan 20 2020 18:45:04.171    9/9      27115      26938      27591
  Jan 20 2020 18:44:49.171    9/9      26878      26539      27143
  Jan 20 2020 18:44:34.171    9/9      26824      26562      27265
  Jan 20 2020 18:44:19.170    9/9      26944      26558      27422
  Jan 20 2020 18:44:06.543    9/9      45218      26512      82600

```

```

Segment-List           : R4
16004

```

```

Delay-Measurement history (uSec):
  Probe Start Timestamp      Pkt (TX/RX)      Average      Min      Max
  Jan 20 2020 18:46:34.174    9/9      26880      26684      27070
  Jan 20 2020 18:46:19.174    9/9      26899      26822      27004
  Jan 20 2020 18:46:04.173    9/9      26813      26571      27164
  Jan 20 2020 18:45:49.172    9/9      26985      26713      27293
  Jan 20 2020 18:45:34.172    9/9      26744      26557      27005
  Jan 20 2020 18:45:19.171    9/9      26740      26435      27093
  Jan 20 2020 18:45:04.171    9/9      27115      26938      27591
  Jan 20 2020 18:44:49.171    9/9      26878      26539      27143
  Jan 20 2020 18:44:34.171    9/9      26824      26562      27265
  Jan 20 2020 18:44:19.170    9/9      26944      26558      27422
  Jan 20 2020 18:44:06.543    9/9      45218      26512      82600

```

```

Atomic path:
Hops           : 127.0.0.0
Delay-Measurement history (uSec):
  Probe Start Timestamp      Pkt (TX/RX)      Average      Min      Max

```

Jan 20 2020 18:46:34.174	3/3	26927	26747	27070
Jan 20 2020 18:46:19.174	3/3	26982	26970	27004
Jan 20 2020 18:46:04.173	3/3	26895	26647	27164
Jan 20 2020 18:45:49.172	3/3	27054	26764	27293
Jan 20 2020 18:45:34.172	3/3	26801	26694	27005
Jan 20 2020 18:45:19.171	3/3	26807	26524	27093
Jan 20 2020 18:45:04.171	3/3	27226	26938	27591
Jan 20 2020 18:44:49.171	3/3	26976	26644	27143
Jan 20 2020 18:44:34.171	3/3	26880	26679	27265
Jan 20 2020 18:44:19.170	3/3	26994	26630	27422
Jan 20 2020 18:44:06.543	3/3	45407	26629	82600

Atomic path:

Hops : 127.0.0.1

Delay-Measurement history (uSec):

Probe Start Timestamp	Pkt (TX/RX)	Average	Min	Max
Jan 20 2020 18:46:34.174	3/3	26865	26705	26988
Jan 20 2020 18:46:19.174	3/3	26846	26822	26881
Jan 20 2020 18:46:04.173	3/3	26787	26581	26939
Jan 20 2020 18:45:49.172	3/3	26954	26728	27180
Jan 20 2020 18:45:34.172	3/3	26724	26577	26957
Jan 20 2020 18:45:19.171	3/3	26705	26452	27032
Jan 20 2020 18:45:04.171	3/3	27043	26972	27124
Jan 20 2020 18:44:49.171	3/3	26848	26550	27062
Jan 20 2020 18:44:34.171	3/3	26800	26562	27204
Jan 20 2020 18:44:19.170	3/3	26927	26576	27327
Jan 20 2020 18:44:06.543	3/3	45128	26521	81961

Atomic path:

Hops : 192.168.0.4

Delay-Measurement history (uSec):

Probe Start Timestamp	Pkt (TX/RX)	Average	Min	Max
Jan 20 2020 18:46:34.174	3/3	26848	26684	26967
Jan 20 2020 18:46:19.174	3/3	26871	26833	26913
Jan 20 2020 18:46:04.173	3/3	26759	26571	26876
Jan 20 2020 18:45:49.172	3/3	26947	26713	27163
Jan 20 2020 18:45:34.172	3/3	26708	26557	26939
Jan 20 2020 18:45:19.171	3/3	26708	26435	27075
Jan 20 2020 18:45:04.171	3/3	27078	27016	27138
Jan 20 2020 18:44:49.171	3/3	26812	26539	27043
Jan 20 2020 18:44:34.171	3/3	26793	26582	27181
Jan 20 2020 18:44:19.170	3/3	26911	26558	27308
Jan 20 2020 18:44:06.543	3/3	45119	26512	81956

Router# **show performance-measurement counters sr-policy name srte_c_10_ep_192.168.0.4**
 Mon Jan 20 18:47:55.499 PST

0/0/CPU0

SR Policy name: srte_c_10_ep_192.168.0.4

Candidate-Path:

Instance	: 2
Preference	: 100
Protocol-origin	: Configured
Discriminator	: 100

Packets:

Total sent	: 141
Total received	: 141

Errors:

Total sent errors	: 0
Total received errors	: 0

Probes:

Total started	: 16
---------------	------

```

Total completed           : 15
Total incomplete         : 0
Total advertisements     : 2
Segment-List             : R4
16004
Packets:
  Total sent              : 141
  Total received         : 141
Errors:
  Total sent errors      : 0
  Total received errors  : 0
Probes:
  Total started          : 16
  Total completed       : 15
  Total incomplete      : 0
  Total advertisements  : 2

```

SR Policy Liveness Monitoring - Hardware Offloading

Table 8: Feature History Table

Feature Name	Release	Description
SR Policy Liveness Monitoring - Hardware Offloading	Release 7.10.1	The liveliness monitoring in Performance Measurement can now be offloaded to the hardware, which is the Network Processing Unit (NPU), on the platform. This feature verifies the liveliness of the path or policy between nodes and helps in optimization, and scalability.

Performance Measurement (PM) hardware offload feature allows the offload of PM transmission (Tx) to the Network Processing Unit (NPU) on the platform, which considerably improves scale and reduces the overall network convergence detection time. This improvement is done by sending rapid failure detection probes (messages) to the routing protocols for recalculating the routing table.

This feature is required in order to quickly react on delay-bound Service Level Agreement (SLAs), for example 5G low-latency, where SRTE policy can quickly re-optimize once the SLA is violated.

Advantages of the PM Hardware Offloading feature are as listed:

- Probes are sent every three milli-seconds
- Complete liveliness of endpoint is now reduced to 10ms from 50ms
- Hardware offloaded probes for liveliness, delay (where headend and endpoint add timestamps), and loss (where endpoint adds counters)
- You can now scale up to 2000 probes per system



CHAPTER 13

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

Topology-Independent Loop-Free Alternate (TI-LFA) uses segment routing to provide link, node, and Shared Risk Link Groups (SRLG) protection in topologies where other fast reroute techniques cannot provide protection.

- Classic Loop-Free Alternate (LFA) is topology dependent, and therefore cannot protect all destinations in all networks. A limitation of LFA is that, even if one or more LFAs exist, the optimal LFA may not always be provided.
- Remote LFA (RLFA) extends the coverage to 90-95% of the destinations, but it also does not always provide the most desired repair path. RLFA also adds more operational complexity by requiring a targeted LDP session to the RLFAs to protect LDP traffic.

TI-LFA provides a solution to these limitations while maintaining the simplicity of the IPFRR solution.

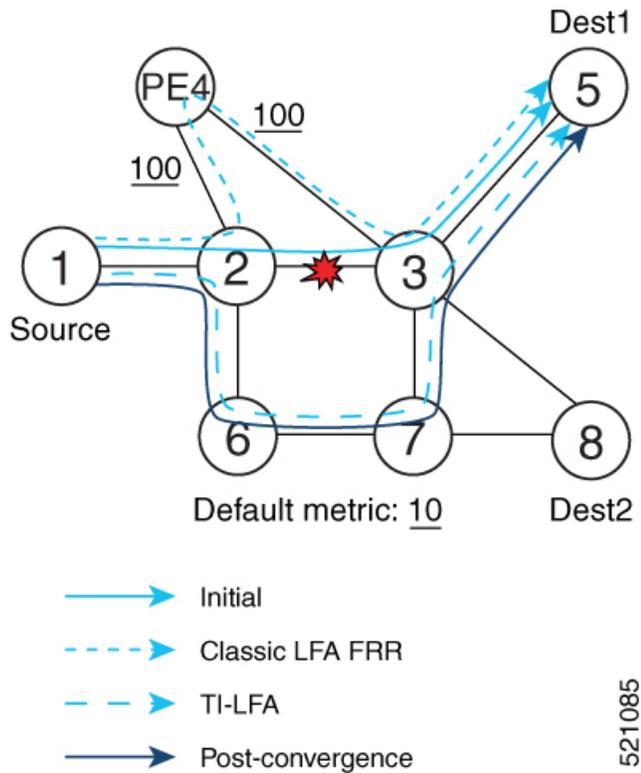
The goal of TI-LFA is to reduce the packet loss that results while routers converge after a topology change due to a link or node failure. Rapid failure repair (< 50 msec) is achieved through the use of pre-calculated backup paths that are loop-free and safe to use until the distributed network convergence process is completed.

The optimal repair path is the path that the traffic will eventually follow after the IGP has converged. This is called the post-convergence path. This path is preferred for the following reasons:

- Optimal for capacity planning — During the capacity-planning phase of the network, the capacity of a link is provisioned while taking into consideration that such link will be used when other links fail.
- Simple to operate — There is no need to perform a case-by-case adjustments to select the best LFA among multiple candidate LFAs.
- Fewer traffic transitions — Since the repair path is equal to the post-convergence path, the traffic switches paths only once.

The following topology illustrates the optimal and automatic selection of the TI-LFA repair path.

Figure 8: 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 Node 2 to Node 5 would be:

Node 2 → Node 6 → Node 7 → Node 3 → Node 5

Node 7 is the PQ-node for destination Node 5. TI-LFA encodes a single segment (prefix SID of Node 7) in the header of the packets on the repair path.

TI-LFA Protection Types

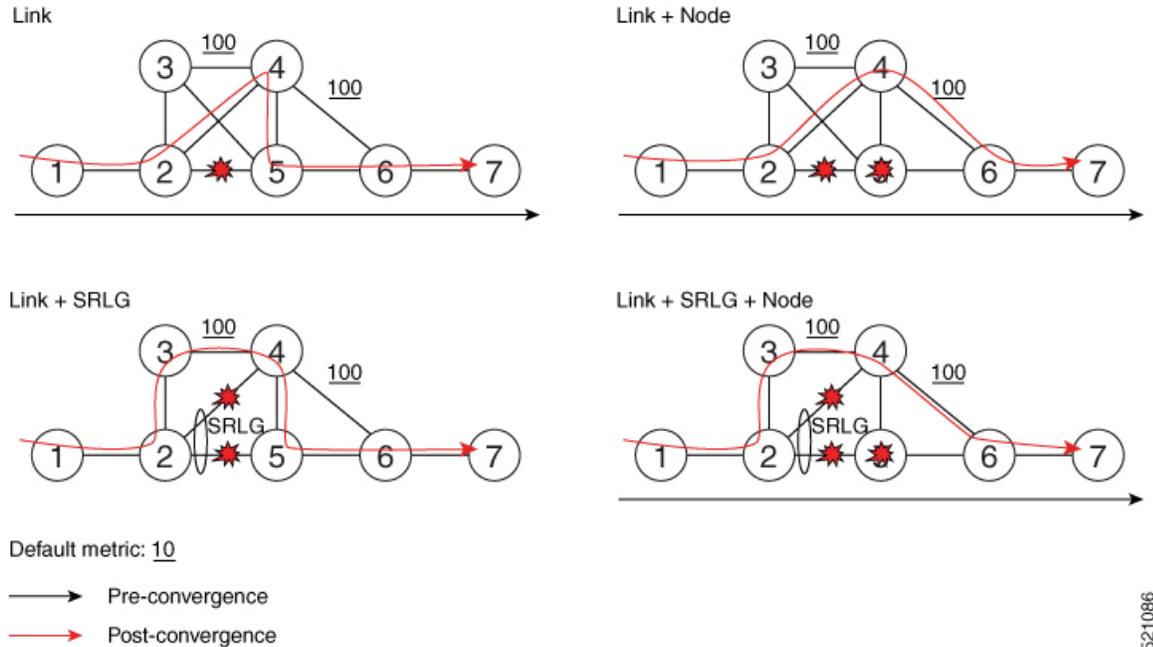
TI-LFA supports the following protection:

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

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

The following example illustrates the link, node, and SRLG protection types. In this topology, Node2 applies different protection models to protect traffic to Node7.

Figure 9: TI-LFA Protection Types



- [Usage Guidelines and Limitations, on page 209](#)
- [Configuring TI-LFA for IS-IS, on page 210](#)
- [Configuring TI-LFA for OSPF, on page 212](#)
- [TI-LFA Node and SRLG Protection: Examples, on page 214](#)
- [Configuring Global Weighted SRLG Protection, on page 215](#)

521086

Usage Guidelines and Limitations

The TI-LFA guidelines and limitations are listed below:

- IGP directly programs a TI-LFA backup path requiring 3 or fewer labels, including the label of the protected destination prefix.
- The platform does not support programming of TI-LFA backup paths requiring more than 3 labels.

TI-LFA Functionality	IS-IS ¹	OSPFv2
<i>Protected Traffic Types</i>		
Protection for SR labeled traffic	Supported	Supported
Protection of IPv4 unlabeled traffic	Supported (IS-ISv4)	Supported

TI-LFA Functionality	IS-IS ¹	OSPFv2
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	Supported	Unsupported
Line Card Disjoint Protection	Supported	Unsupported
Interface Types		
Ethernet Interfaces	Supported	Supported
TI-LFA with L3VPN	Supported	Supported
Ethernet Bundle Interfaces	Supported	Supported
TI-LFA over GRE Tunnel as Protecting Interface	Unsupported	Unsupported
Additional Functionality		
Maximum number of labels that can be pushed on the backup path (including the label of the protected prefix)	3	3
BFD-triggered	Supported	Supported
BFDv6-triggered	Supported	N/A
Prefer backup path with lowest total metric	Supported	Supported
Prefer backup path from ECMP set	Supported	Supported
Prefer backup path from non-ECMP set	Supported	Supported
Load share prefixes across multiple backups paths	Supported	Supported
Limit backup computation up to the prefix priority	Supported	Supported

¹ Unless specified, IS-IS support is IS-ISv4 and IS-ISv6

Configuring TI-LFA for IS-IS

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

Before you begin

Ensure that the following topology requirements are met:

- Routers are configured with IS-IS.

- Segment routing for IS-IS is configured. See [Enabling Segment Routing for IS-IS Protocol, on page 41](#).

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration 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.

	Command or Action	Purpose
Step 6	fast-reroute per-prefix ti-lfa Example: <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa</pre>	Enables per-prefix TI-LFA fast reroute link protection.
Step 7	fast-reroute per-prefix tiebreaker { node-protecting srlg-disjoint } index priority Example: <pre>RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tie-breaker srlg-disjoint index 100</pre>	Enables TI-LFA node or SRLG protection and specifies the tiebreaker priority. Valid <i>priority</i> values are from 1 to 255. The lower the <i>priority</i> value, the higher the priority of the rule. Link protection always has a lower priority than node or SRLG protection. Note The same attribute cannot be configured more than once on an interface. Note For IS-IS, TI-LFA node protection and SRLG protection can be configured on the interface or the instance.

TI-LFA has been successfully configured for segment routing.

Configuring TI-LFA for OSPF

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



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

Before you begin

Ensure that the following topology requirements are met:

- Routers are configured with OSPF.
- Segment routing for OSPF is configured. See [Enabling Segment Routing for OSPF Protocol, on page 61](#).

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**

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# configure	Enters global configuration mode.
Step 2	router ospf process-name 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 area-id Example: RP/0/RP0/CPU0:router(config-ospf)# area 1	Enters area configuration mode.
Step 4	interface type interface-path-id 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.
Step 7	fast-reroute per-prefix tiebreaker {node-protecting srlg-disjoint} index priority Example: RP/0/RP0/CPU0:router(config-ospf-ar-if)# fast-reroute per-prefix tie-breaker srlg-disjoint index 100	Enables TI-LFA node or SRLG protection and specifies the tiebreaker priority. Valid <i>priority</i> values are from 1 to 255. The higher the <i>priority</i> value, the higher the priority of the rule. Link protection always has a lower priority than node or SRLG protection. Note The same attribute cannot be configured more than once on an interface.

TI-LFA has been successfully configured for segment routing.

TI-LFA Node and SRLG Protection: Examples

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

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

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

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

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

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

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

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

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

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

Configuring Global Weighted SRLG Protection

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

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

The administrative weight (cost) of the SRLG can be configured using the **admin-weight** command. This command can be applied for all SRLG (global), or for a specific (named) SRLG. The default (global) admin-weight value is 1 for IS-IS.

Configuration Examples: Global Weighted SRLG Protection for IS-IS

There are three types of configurations that are supported for the global weighted SRLG protection feature for IS-IS:

- Local SRLG with global weighted SRLG protection
- Remote SRLG flooding
- Remote SRLG static provisioning

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

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg)# exit
RP/0/RP0/CPU0:router(config-srlg)# name group1 value 100
RP/0/RP0/CPU0:router(config-srlg)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-af)# fast-reroute per-prefix srlg-protection weighted-global
RP/0/RP0/CPU0:router(config-isis-af)# fast-reroute per-prefix tiebreaker srlg-disjoint index
1
RP/0/RP0/CPU0:router(config-isis-af)# exit
RP/0/RP0/CPU0:router(config-isis)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis-if-af)# exit
RP/0/RP0/CPU0:router(config-isis-if)# exit
RP/0/RP0/CPU0:router(config-isis)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000
```

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

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

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

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

```
RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# name group1 value 100
RP/0/RP0/CPU0:router(config-srlg)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-af)# advertise application lfa link-attributes srlg
```

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

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

```
RP/0/RP0/CPU0:router(config-isis-af)# exit
RP/0/RP0/CPU0:router(config-isis)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis-if-af)# exit
RP/0/RP0/CPU0:router(config-isis-if)# exit
RP/0/RP0/CPU0:router(config-isis)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.1 next-hop ipv4
address 10.0.4.2
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.2 next-hop ipv4
address 10.0.4.1
```




CHAPTER 14

Configure Segment Routing Microloop Avoidance

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

- [About Segment Routing Microloop Avoidance, on page 219](#)
- [Usage Guidelines and Limitations, on page 221](#)
- [Configure Segment Routing Microloop Avoidance for IS-IS, on page 222](#)
- [Configure Segment Routing Microloop Avoidance for OSPF, on page 223](#)

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, link up, or metric change events). 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 resolves 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 to steer the traffic to that destination loop-free over the post-convergence path.

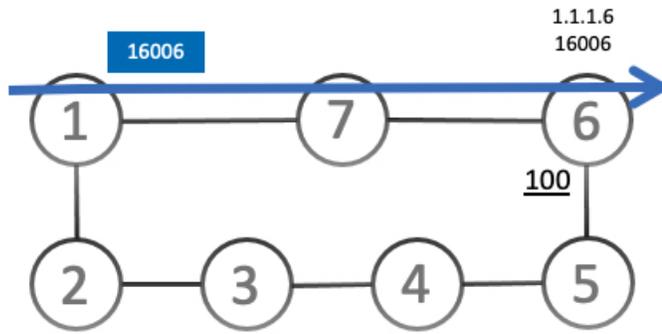
The IGP updates the forwarding table and temporarily (based on a RIB update delay timer) installs the SID-list imposition entries associated with the microloop-avoidant path for the destination with possible microloops.

After the RIB update delay timer expires, IGP updates the forwarding table, removing the microloop-avoidant SID list and 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.

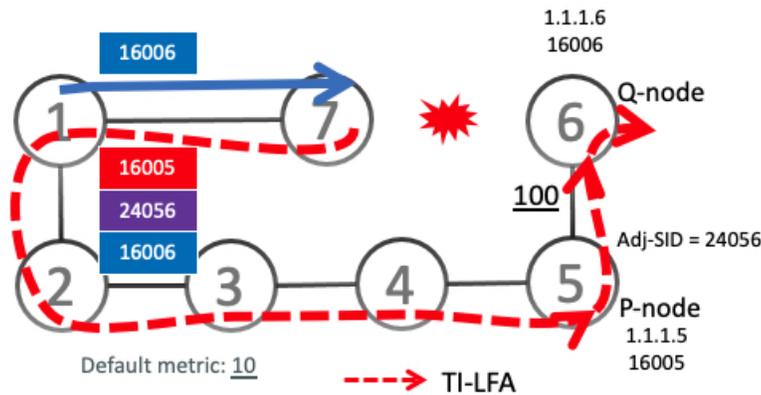
In the topology below, microloops can occur after the failure of the link between Node6 and Node7.

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

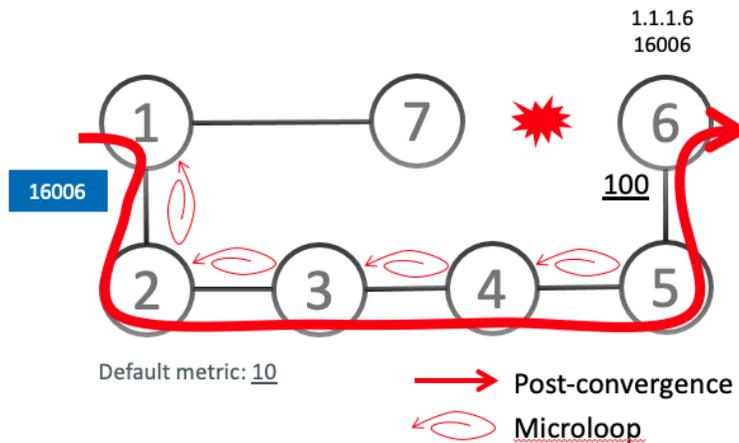


Default metric: 10

TI-LFA on Node7 pre-computes a backup path for traffic to Node6 (prefix SID 16006) that will be activated if the link between Node7 and Node6 goes down. In this network, the backup path would steer traffic toward Node5 (prefix SID 16005) and then via link between Node5 and Node6 (adj-SID 24056). All nodes are notified of the topology change due to the link failure.



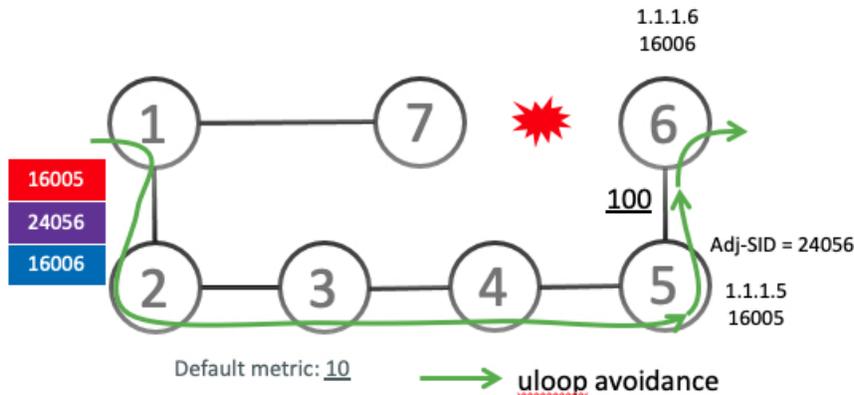
However, if nodes along the path do not converge at the same time, microloops can be introduced. For example, if Node2 converged before Node3, Node3 would send traffic back to Node2 as the shortest IGP path to Node6. The traffic between Node2 and Node3 creates a microloop.



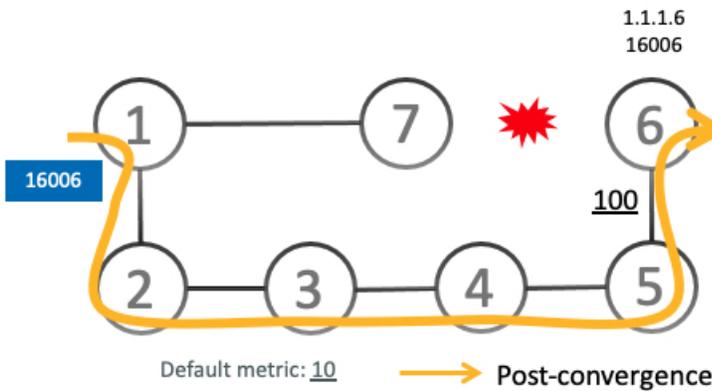
With microloop avoidance configured on Node1, a post-convergence path is computed and possible microloops on the post-convergence path for any destination are detected.

If microloops are possible on the post-convergence path to Node6, a microloop-avoidant path is constructed to steer the traffic to Node6 loop-free over the microloop-avoidant path {16005, 24056, 16006}.

Node1 updates the forwarding table and installs the SID-list imposition entries for those destinations with possible microloops, such as Node6. All nodes converge and update their forwarding tables, using SID lists where needed.



After the RIB update delay timer expires, the microloop-avoidant path is replaced with regular forwarding paths; traffic now natively follows the post-convergence path.



Usage Guidelines and Limitations

IGP directly programs a microloop-avoidant path requiring 3 or fewer labels, including the label of the destination prefix.

The platform does not support programming of microloop-avoidant paths requiring more than 3 labels.

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 41](#).

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/RP0/CPU0:router# <code>configure</code>	Enters global configuration 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.

	Command or Action	Purpose
Step 5	microloop avoidance rib-update-delay <i>delay-time</i> Example: <pre>RP/0/RP0/CPU0:router(config-isis-af) # microloop avoidance rib-update-delay 3000</pre>	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 61](#).

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: <pre>RP/0/RP0/CPU0:router# configure</pre>	Enters global configuration mode.
Step 2	router ospf <i>process-name</i> Example: <pre>RP/0/RP0/CPU0:router(config)# router ospf 1</pre>	Enables OSPF routing for the specified routing process, and places the router in router configuration mode.
Step 3	microloop avoidance segment-routing Example: <pre>RP/0/RP0/CPU0:router(config-ospf)# microloop avoidance segment-routing</pre>	Enables Segment Routing Microloop Avoidance.

	Command or Action	Purpose
Step 4	<p>microloop avoidance rib-update-delay <i>delay-time</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ospf)# microloop avoidance rib-update-delay 3000</pre>	<p>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.</p>



CHAPTER 15

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 225](#)
- [Segment Routing and LDP Interoperability, on page 226](#)
- [Configuring Mapping Server, on page 229](#)
- [Enable Mapping Advertisement, on page 231](#)
- [Enable Mapping Client, on page 233](#)

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

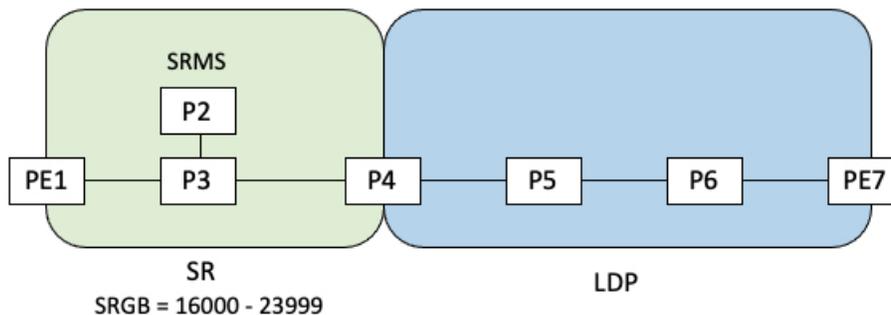
IGP provides mechanisms through which segment routing (SR) interoperates with label distribution protocol (LDP). The control plane of segment routing co-exists with LDP.

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.

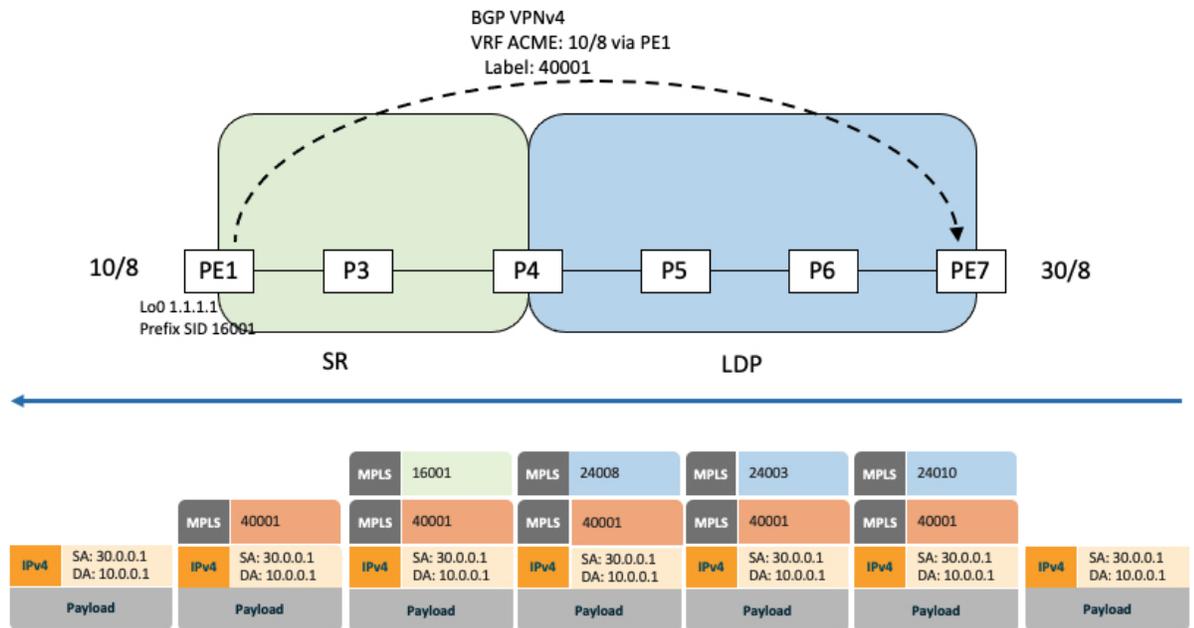


In this mixed network:

- Nodes PE1, P2, P3, and P4 are SR-capable
- Nodes P4, P5, P6, and PE7 are LDP-capable
- Nodes PE1, P2, P3, and P4 are configured with segment routing global block (SRGB) range of 16000 to 23999
- Nodes PE1, P2, P3, and P4 are configured with node segments of 16001, 16002, 16003, and 16004 respectively

A service flow must be established from PE1 to PE3 over a continuous MPLS tunnel. This requires SR and LDP to interoperate.

LDP-to-SR Traffic Direction



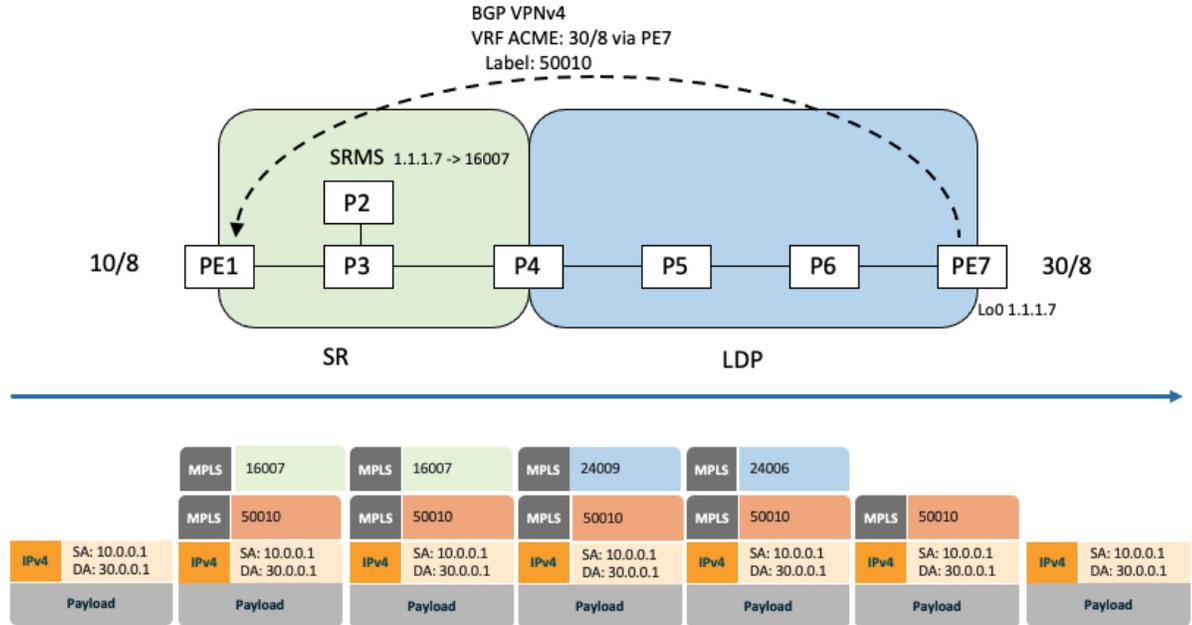
The traffic flow in the LDP-to-SR direction involves the following:

1. PE7 learns a service route with service label 40001 and BGP nhop PE1.
2. PE7 has an LDP label binding (24010) from the nhop P6 for the FEC PE1. PE7 forwards the packet to P6.
3. P6 has an LDP label binding (24003) from its nhop P5 for the FEC PE1. P6 forwards the packet to P5.
4. P5 has an LDP label binding (24008) from its nhop P4 for the FEC PE1. P5 forwards the packet to P4.
5. P4 does not have an LDP binding from its nhop P3 for the FEC PE1. But P4 has an SR node segment to the IGP route PE1. P4 forwards the packet to P3 and swaps its local LDP label (24008) for FEC PE1 by the equivalent node segment 16001. This process is called label merging.
6. P3 pops 16001, assuming PE1 has advertised its node segment 16001 with the penultimate-pop flag set and forwards to PE1.

7. PE1 receives the packet and processes the service label.

The end-to-end MPLS LSP is established from an LDP LSP from PE7 to P4 and the related node segment from P4 to PE1.

SR-to-LDP Traffic Direction



Suppose that the operator configures P2 as a Segment Routing Mapping Server (SRMS) and advertises the mappings (1.1.1.7, 16007 for PE7). Because PE7 is non-SR capable, the operator configures that mapping policy at the SRMS; the SRMS advertises the mapping on behalf of the non-SR capable nodes. Multiple SRMS servers can be provisioned in a network for redundancy. The mapping server advertisements are only understood by the SR-capable nodes. The SR-capable routers install the related node segments in the MPLS data plane in exactly the same manner as if node segments were advertised by the nodes themselves.

The traffic flow in the SR to LDP direction involves the following:

1. PE1 learns a service route with service label 50010 and BGP nhop PE7.
2. PE1 has an SR label binding (16007) learned from the SRMS (P2) for PE7.
3. PE1 installs the node segment 16007 following the IGP shortest-path with nhop P3.
4. P3 swaps 16007 for 16007 and forwards to P4.
5. The nhop for P4 for the IGP route PE7 is non-SR capable, since P5 does not advertise the SR capability. However, P4 has an LDP label binding from that nhop for the same FEC (for example, LDP label 24009). P4 would then swap 16007 for 24009 and forward to P5. We refer to this process as label merging.
6. P5 swaps this label with the LDP label received from P6 (for example, LDP label 24006) and forwards to P6.
7. P6 pops the LDP label and forwards to PE7.
8. PE7 receives the packet and processes the service label.

The end-to-end MPLS LSP is established from an SR node segment from PE1 to P4 and an LDP LSP from P4 to PE7.

Observe that the capabilities provided by the SRMS are only required in the SR-to-LDP direction.

Configuring Mapping Server

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

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 global configuration 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 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, IGP prefix and Flexible Algorithm SIDs, and Nil-FEC (forwarding equivalence classes) LSP Ping and Traceroute functionality.

- [MPLS Ping and Traceroute for BGP and IGP Prefix-SID, on page 235](#)
- [Examples: MPLS Ping, Traceroute, and Tree Trace for Prefix-SID, on page 236](#)
- [MPLS LSP Ping and Traceroute Nil FEC Target, on page 238](#)
- [Examples: LSP Ping and Traceroute for Nil_FEC Target, on page 238](#)
- [Segment Routing Ping and Traceroute, on page 240](#)
- [Segment Routing Ping and Traceroute for Flexible Algorithm, on page 244](#)
- [Segment Routing over IPv6 OAM, on page 245](#)

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

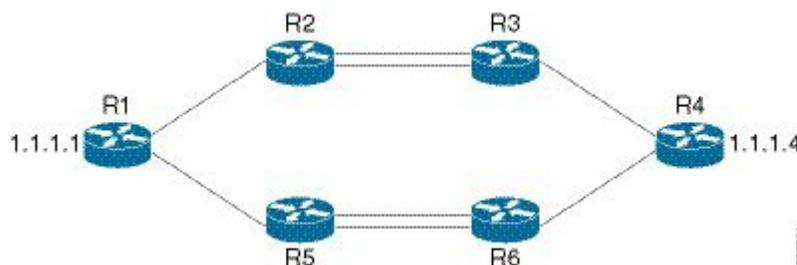
Live (TTL) value of the packet that carries the echo request. When the MPLS echo request message hits a transit node, it checks the TTL value and if it is expired, the packet is passed to the control plane, else the message is forwarded. If the echo message is passed to the control plane, a reply message 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   Next Hop    Switched
-----
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 Ping and Traceroute for Flexible Algorithm

Flexible Algorithm validation method is based on segment identifier (SID) label and label switched path (LSP) destination, instead of being based on IP address. The assigner is validated against the topology prefix information provided by SR-PCE database. If the assigner is valid, then the label given is also validated against the SR-PCE database. On the egress side, the destination label is contained in a new SR Label sub-TLV. This label is verified against a SID list provided by SR-PCE.



Note Observe the following guidelines and restrictions:

- All routers within an area must share the same Flexible Algorithm definition for a Flexible Algorithm to be valid.
- All routers within the domain must be configured with the same SRGB range of values.
- BGP-LS must be enabled.
- Only prefix SIDs and Flexible Algorithm SIDs are supported.
- Only single label stack is supported.

Segment Routing Ping for Flexible Algorithm

```
Router# ping sr-mpls labels 16131 lsp-end-point 10.1.1.5
Fri Dec 13 19:26:29.517 IST
```

```

Sending 5, 100-byte MPLS Echos with SR Label FEC with lsp end point 10.1.1.5, SID Label(s)
[16131],
    timeout is 2 seconds, send interval is 0 msec:

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

Type escape sequence to abort.

!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/6 ms

```

Segment Routing Traceroute for Flexible Algorithm

```

Router# traceroute sr-mpls labels 16130 lsp-end-point 10.1.1.5
Fri Dec 13 19:26:59.368 IST

Tracing MPLS Label Switched Path to SR Label FEC with lsp end point 10.1.1.5, SID Label(s)
[16130], timeout is 2 seconds

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

Type escape sequence to abort.

 0 13.13.13.1 MRU 1500 [Labels: 16130 Exp: 0]
L 1 13.13.13.3 MRU 1500 [Labels: 16130 Exp: 0] 5 ms
L 2 16.16.16.4 MRU 1500 [Labels: implicit-null Exp: 0] 4 ms
! 3 18.18.18.5 4 ms

```

Segment Routing over IPv6 OAM

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

Restrictions and Usage Guidelines

The following restriction applies for SRv6 OAM:

- Ping to an SRv6 SID is not supported.

Examples: SRv6 OAM

The following example shows using ping in an SRv6 network.

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

The following example shows using traceroute in an SRv6 network.

```
RP/0/RP0/CPU0:Router# traceroute ipv6 2001::33:33:33:33 probe 1 timeout 0 srv6
Fri Sep 14 15:59:25.170 UTC
Type escape sequence to abort.
Tracing the route to 2001::33:33:33:33
 1 2001::22:22:22:22[IP tunnel: DA=cafe:0:0:a4:1::: SRH =(2001::33:33:33:33 ,SL=1)] 2
msec
 2 2001::2:2:2:2[IP tunnel: DA=cafe:0:0:a4:1::: SRH =(2001::33:33:33:33 ,SL=1)] 2 msec
 3 2001::44:44:44:44 2 msec
 4 2001::33:33:33:33 3 msec
```

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

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

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

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

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

```
RP/0/RP0/CPU0:Router# traceroute 10.15.15.1 vrf red
Mon Sep 17 20:07:18.478 UTC

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

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

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

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