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.

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
Local ECMP Support on SRTE	Release 7.10.1	This feature supports the top security identifier (SID) resolved route interior gateway protocol (IGP) path's weight to be used in relative weight computation of outgoing paths of the SRTE policy. Now ECMP supports SRTE and computes per-egress-path relative weight using underlay (IGP) weight from Forwarding Information Base (FIB) / Routing Information Base (RIB) in addition to Segment List (SL) weight and number of primary egress paths for SL. Thus, IGP load balancing required at individual paths is also maintained. This feature is enabled by default and no configuration changes are required.
Segment Routing Traffic Engineering (SR-TE)	Release 7.5.2	You create a policy to steer traffic between a source-and-destination pair using the concept of source routing, where the source calculates the path and encodes it in the packet header as a segment. This functionality uses a single intelligent source and does not rely on the remaining nodes to compute a path through the network. This feature utilizes network bandwidth more effectively than traditional MPLS-TE networks by using ECMP at every segment level. Cisco 8000 series routers support the imposition of up to 8 MPLS transport labels, including TI-LFA backup labels for the protection of the top SID of an SR policy.

SR-TE Policy Overview

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

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

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

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

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

Usage Guidelines and Limitations

Observe the following guidelines and limitations for the platform.

- Broadcast links are not supported, configure IGP's interface as P2P (point-to-point).
- Before configuring SR-TE policies, use the **distribute link-state** command under IS-IS or OSPF to distribute the link-state database to external services.
- GRE tunnel as primary interface for an SR policy is not supported.
- GRE tunnel as backup interface for an SR policy with TI-LFA protection is not supported.
- Head-end computed inter-domain SR policy with Flex Algo constraint and IGP redistribution is not supported.
- The number of segment-lists (SLs) per SR policy is limited to a maximum of seven. If you need a policy with more than seven segment-lists, perform the following.
  1. Delete the existing SR policies.
  2. Configure the following command:

     ```
     Router(config)#hw-module profile cef te-tunnel highscale-no-ldp-over-te
     ```
  3. Reload the router for the configuration to take effect.
  4. Configure SR policy with more than seven segment-lists.

Note If you configure this command, the Autoroute feature will not work.

- For SR over SR policy traffic, the hardware can manage traffic accounting either for the SR label or SR policy but not for both. By default, the accounting of SR over SRTE traffic happens on SR policy. Perform the following steps to manage traffic using SR label accounting instead of SR policy accounting:

  1. Configure the following command:

     ```
     Router(config)#hw-module profile cef label-over-te-counters
     ```

  2. Reload the router for the configuration to take effect.

The following features are supported for SR-TE:

- SR-TE On-Demand Next Hop/Automated Steering (ODN/AS) is supported for global IPv4 BGP and IPv6 BGP prefixes with colors
- SR-TE BSID-based AS
- SR-TE head-end path computation
- LFA at SR-TE head-end
- Per-SR policy BSID label counters
- Per-SR policy aggregate counters
- Per-SR policy, per-segment list aggregate counters
- Per-SR policy, per-segment list, per-protocol aggregate counters:
  - Unlabeled IP – Unlabeled IPv4 and IPv6 traffic steered over SR policy
  - Labeled MPLS – Labeled traffic with BSID as top of label stack steered over SR policy
- Supports PCEP at SR-TE head-end
- Supports TI-LFA at SR-TE head-end
- Supports a maximum SID Depth (MSD) of 8 MPLS labels. In case it exceeds 8 MPLS labels, backup path is not created.
- SR-TE policy with autoroute-include based steering.
- Support for a BGP service route with a combination of paths pointing to SR native LSP with protection and paths pointing to SR-TE/BSID.

The following features are not supported:

- SR-TE ODN/AS for 6PE, VPNv4, VPNv6 (6vPE), EVPN
- SR-TE per-flow policy (PFP)
- Static Route Traffic-Steering using SR-TE Policy
- LDP over SR-TE Policy

- Mix of BSID and native paths with protection
- IPv6 IGP Routes over SRTE Policy

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:

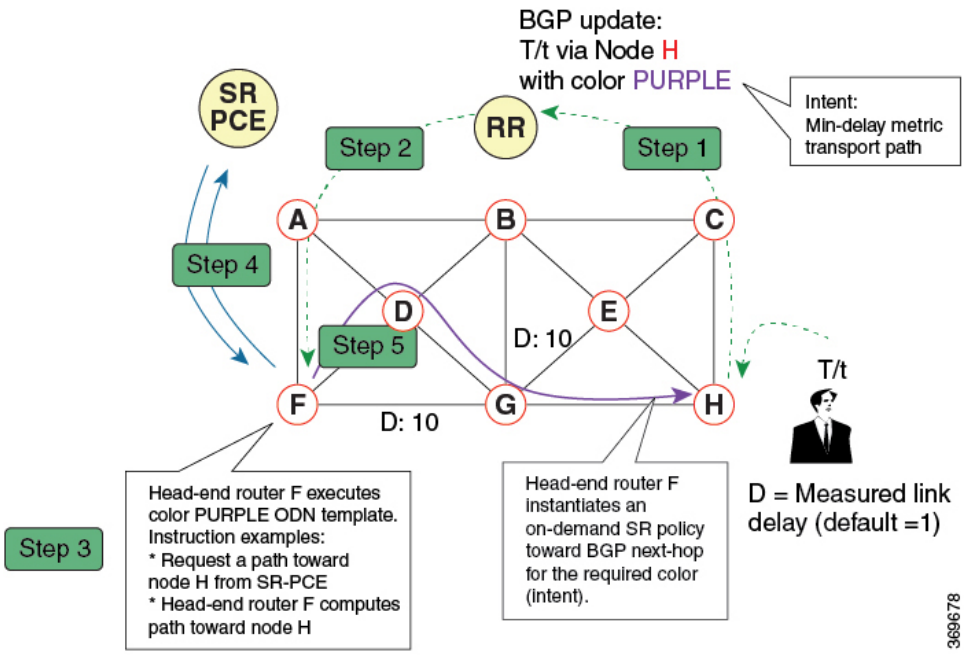
- On-Demand SR Policy – SR On-Demand Next-Hop , on page 4
- Manually Provisioned SR Policy, on page 16

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)

SR-ODN Configuration Steps

Note In Cisco IOS XR release 7.5.2, SR-ODN configurations with Flexible Algorithm constraints can be configured using either of the following commands:

- **segment-routing traffic-eng on-demand color** *color* **dynamic sid-algorithm** *algorithm-number*
- **segment-routing traffic-eng on-demand color** *color* **constraints segments sid-algorithm** *algorithm-number*

Starting with Cisco IOS XR release 7.9.1, the **dynamic sid-algorithm** *algorithm-number* command has been deprecated. Only the **constraints segments sid-algorithm** *algorithm-number* command is supported. Configurations stored in NVRAM using the **dynamic sid-algorithm** *algorithm-number* command will be rejected at boot-up.

As a result, SR-ODN configurations with Flexible Algorithm constraints using the **dynamic sid-algorithm** *algorithm-number* CLI must be re-configured using the **constraints segments sid-algorithm** *algorithm-number* CLI.

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.

   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, on page 36.

2. Define BGP color extended communities. Refer to the "Implementing BGP" chapter in the *BGP Configuration Guide for Cisco 8000 Series Routers*.

3. Define routing policies (using routing policy language [RPL]) to set BGP color extended communities. Refer to the "Implementing Routing Policy" chapter in the *Routing Configuration Guide for Cisco 8000 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 Cisco 8000 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 **constraints segments sid-algorithm** *algorithm-number* command to configure the SR Flexible Algorithm constraints. The *algorithm-number* range is from 128 to 255.

  ```
  Router(config-sr-te-color)# constraints segments sid-algorithm 128
  ```

Configuring SR-ODN: Examples

Configuring SR-ODN: Layer-3 Services Examples

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
SR-ODN: Layer 3 IPv6 BGP Services	Release 7.9.1	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). This feature introduces support for Layer 3 IPv6 BGP services (VPNv4/VPNv6) over SR-ODN.

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 four 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
  192.68.0.0/24
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
```

L3VPN IPv4 Services: Verifying BGP VRF Information

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

```
Router# show bgp vrf vrf_cust1

BGP VRF vrf_cust1, state: Active
BGP Route Distinguisher: 1.1.1.4:101
VRF ID: 0x60000007
BGP router identifier 1.1.1.4, 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: 1.1.1.4:101 (default for vrf vrf_cust1)
*> 44.1.1.0/24        40.4.101.11                            0 400 {1} i
*>i55.1.1.0/24        1.1.1.5                      100       0 500 {1} i
*>i88.1.1.0/24        1.1.1.8 C:10                 100       0 800 {1} i
*>i99.1.1.0/24        1.1.1.9                      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.

```
Router# show bgp vrf vrf_cust1 88.1.1.0/24

BGP routing table entry for 88.1.1.0/24, Route Distinguisher: 1.1.1.4:101
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker               282         282
Last Modified: May 20 09:23:34.112 for 00:06:03
Paths: (1 available, best #1)
  Advertised to CE peers (in unique update groups):
    40.4.101.11
  Path #1: Received by speaker 0
  Advertised to CE peers (in unique update groups):
    40.4.101.11
  800 {1}
    1.1.1.8 C:10 (bsid:24036) (metric 20) from 1.1.1.55 (1.1.1.8)
      Received Label 24012
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate,
imported
      Received Path ID 0, Local Path ID 1, version 273
      Extended community: Color:10 RT:100:1
      Originator: 1.1.1.8, Cluster list: 1.1.1.55
      SR policy color 10, up, registered, bsid 24036, if-handle 0x08000024

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

L3VPN IPv4 Services: 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 198.51.100.1/24 points to the BSID label corresponding to an SR policy. Other non-colored prefixes, such as 192.0.2.2/24, point to BGP next-hop.

```
Router# show cef vrf vrf_cust1

Prefix              Next Hop            Interface
------------------- ------------------- ------------------
0.0.0.0/0           drop                default handler
0.0.0.0/32          broadcast
10.0.0.1/8          attached            TenGigE0/0/0/0.101
172.16.0.1/12       broadcast           TenGigE0/0/0/0.101
172.16.0.2/12       receive             TenGigE0/0/0/0.101
172.16.0.3/12       172.16.0.3/12       TenGigE0/0/0/0.101
172.16.0.4/12       broadcast           TenGigE0/0/0/0.101
192.168.0.1/16      172.16.0.3/12       <recursive>
192.0.2.2/24        10.0.0.2/8          <recursive>
198.51.100.1/24     24036 (via-label)   <recursive>
```

The following output displays CEF details for prefix 198.51.100.1/24. Note that the prefix is associated with an SR policy with BSID value of 24036.

```
Router# show cef vrf vrf_cust1 198.51.100.1/24

198.51.100.1/24, 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}
```

L3VPN IPv4 Services: 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 10.0.0.8.

```
Router# show segment-routing traffic-eng policy color 10 tabular

 Color             Endpoint  Admin   Oper              Binding
                             State  State                 SID
------ -------------------- ------ ------ --------------------
    10              10.0.0.8     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.

```
Router# show segment-routing traffic-eng policy binding-sid 24036

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

Color: 10, End-point: 10.0.0.8
  Name: srte_c_10_ep_10.0.0.8
  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_10.0.0.8_discr_200
        PLSP-ID: 12
      Dynamic (valid)
        Metric Type: TE,   Path Accumulated Metric: 30
            16009 [Prefix-SID, 10.0.0.9]
            16008 [Prefix-SID, 10.0.0.8]
    Preference: 100 (BGP ODN)
      Requested BSID: dynamic
      PCC info:
        Symbolic name: bgp_c_10_ep_10.0.0.8_discr_100
        PLSP-ID: 11
      Dynamic (pce 10.0.0.57) (valid)
        Metric Type: TE,   Path Accumulated Metric: 30
            16009 [Prefix-SID, 10.0.0.9]
            16008 [Prefix-SID, 10.0.0.8]
  Attributes:
    Binding SID: 24036
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

L3VPN IPv4 Services: 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 10.0.0.8.

```
Router# show segment-routing traffic-eng forwarding policy binding-sid 24036 tabular

Color Endpoint        Segment      Outgoing Outgoing     Next Hop     Bytes        Pure
                      List         Label    Interface                Switched     Backup
----- --------------- ------------ -------- ------------ ------------ ------------ ------
10    10.0.0.8        dynamic      16009    Gi0/0/0/4    10.4.5.5     0
                                   16001    Gi0/0/0/5    10.5.8.8     0            Yes

Router# 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: 10.0.0.8
  Name: srte_c_10_ep_10.0.0.8
  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: 10.5.8.8
          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
```

L3VPN IPv6 Services: Verifying BGP VRF Information

Use the **show bgp vrf** command to display BGP prefix information for VRF instances.

The following output displays the details for prefix 2020:0:0:1::. 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 51006.

```
Router# show bgp vrf vrf_cust1 ipv6 unicast 2020:0:0:1::

BGP routing table entry for 2020:0:0:1::/64, Route Distinguisher: 100:1001
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker               66662        66662
Last Modified: Mar 16 09:18:40.678 for 00:01:36
Paths: (1 available, best #1)
```

```
  Advertised to CE peers (in unique update groups):
    2002::6701:b02
  Path #1: Received by speaker 0
  Advertised to CE peers (in unique update groups):
    2002::6701:b02
  2001
    100.2.1.1 C:10 (bsid:51006) (metric 40) from 100.2.1.1 (100.2.1.1)
      Received Label 51000
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate,
imported
      Received Path ID 1, Local Path ID 1, version 49971
      Extended community: Color[CO-Flag]:10[10] RT:100:1
      SR policy color 10, up, registered, bsid 51006, if-handle 0x0f000c9c

     Source AFI: VPNv6 Unicast, Source VRF: default, Source Route Distinguisher: 100:2001[0m
```

L3VPN IPv6 Services: Verifying Forwarding (CEF) Table

Use the **show cef vrf** command to display the contents of the CEF table for the VRF instance.

The following output displays CEF details for prefix 2020:0:0:1::/64. Note that the prefix is associated with an SR policy with BSID value of 51006.

```
Router# show cef vrf vrf_cust1 ipv6 2020:0:0:1:: detail

2020:0:0:1::/64, version 229, internal 0x5000001 0x30 (ptr 0xba4b9050) [1], 0x0 (0x0), 0x208
 (0x9a6e3858)
 Updated Mar 16 09:18:41.170
 Prefix Len 64, traffic index 0, precedence n/a, priority 3
  gateway array (0xbc2fe5d0) reference count 16, flags 0x2038, source rib (7), 0 backups
                [1 type 1 flags 0x48441 (0xbe849048) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Mar 16 09:18:41.170
 LDI Update time Mar 16 09:18:41.170
   via local-label 51006, 3 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x9a86d0e0 0x0]
    recursion-via-label
    next hop VRF - 'default', table - 0xe0000000
    next hop via 51006/0/21
     labels imposed {51000}

    Load distribution: 0 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   recursive                 51006/0        [0m
```

L3VPN IPv6 Services: Verifying SR Policy

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

```
Router# show segment-routing traffic-eng policy endpoint ipv4 100.2.1.1 color 10

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

Color: 10, End-point: 100.2.1.1
  Name: srte_c_10_ep_100.2.1.1
  Status:
    Admin: up  Operational: up for 00:01:21 (since Mar 16 09:18:41.096)
  Candidate-paths:
    Preference: 200 (BGP ODN) (active)
```

```
    Requested BSID: dynamic
    Constraints:
      Protection Type: protected-preferred
      Maximum SID Depth: 8
    Dynamic (valid)
      Metric Type: TE,   Path Accumulated Metric: 30
        SID[0]: 51002 [Adjacency-SID, 101.1.3.1 - 101.1.3.2]
        SID[1]: 41304 [Adjacency-SID, 101.3.4.1 - 101.3.4.2]
        SID[2]: 41400 [Adjacency-SID, 101.2.4.2 - 101.2.4.1]
  Preference: 100 (BGP ODN) (inactive)
    Requested BSID: dynamic
    PCC info:
      Symbolic name: bgp_c_10_ep_100.2.1.1_discr_100
      PLSP-ID: 3
    Constraints:
      Protection Type: protected-preferred
      Maximum SID Depth: 8
    Dynamic (pce) (inactive)
      Metric Type: NONE,   Path Accumulated Metric: 0
Attributes:
  Binding SID: 51006
  Forward Class: Not Configured
  Steering labeled-services disabled: no
  Steering BGP disabled: no
  IPv6 caps enable: yes
  Invalidation drop enabled: no
  Max Install Standby Candidate Paths: 0
```

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 19 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 for Traffic Management section in the *Configure Segment Routing Path Computation Element* chapter.

Cumulative Metric Bounds (Delay-Bound Use-Case)

SRTE can calculate a shortest path with cumulative metric bounds. For example, consider these metric bounds:

- IGP metric <= 10

- TE metric <= 60
- Hop count <= 4
- Latency <= 55

When an SR policy is configured on a head-end node with these metric bounds, a path is finalized towards the specified destination only if it meets each of these criteria.

You can set the maximum number of attempts for computing a shortest path that satisfies the cumulative metric bounds criteria, by using the **kshortest-paths** command in SR-TE configuration mode.

Restrictions

- PCE-based cumulative metric bounds computations are not supported. You must use non-PCE (SR-TE topology) based configuration for path calculation, for cumulative bounds.
- If you use PCE dynamic computation configuration with cumulative bounds, the PCE computes a path and validates against cumulative bounds. If it is valid, then the policy is created with this path on PCC. If the initial path doesn't respect the bounds, then the path is not considered, and no further K-shortest path algorithm is executed to find the path.

Configuring SRTE Shortest Path Calculation For Cumulative Metric Bounds

You can enable this feature for SR, and ODN SR policy configurations, as shown below.

SR Policy

SR Policy - A policy called **fromAtoB_XTC** is created towards destination IP address 192.168.0.2. Also, the candidate-paths preference, and other attributes are enabled.

```
Router# configure terminal
Router(config)# segment-routing traffic-eng policy fromAtoB_XTC
Router(config-sr-te-policy)# color 2 end-point ipv4 192.168.0.2
Router(config-sr-te-policy)# candidate-paths preference 100
Router(config-sr-te-policy-path-pref)# dynamic metric type te
```

Cumulative Metric bounds – IGP, TE, hop count, and latency metric bounds are set. SRTE calculates paths only when each criterion is satisfied.

```
Router(config-sr-te-policy-path-pref)# constraints bounds cumulative
Router(config-sr-te-pref-const-bounds-type)# type igp 10
Router(config-sr-te-pref-const-bounds-type)# type te 60
Router(config-sr-te-pref-const-bounds-type)# type hopcount 4
Router(config-sr-te-pref-const-bounds-type)# type latency 55
Router(config-sr-te-pref-const-bounds-type)# commit
```

ODN SR Policy

SR ODN Policy – An SR ODN policy with color 1000 is created. Also, the candidate-paths value is on-demand.

```
Router# configure terminal
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# on-demand color 1000 dynamic metric type te
Router(config-sr-te)# candidate-paths on-demand
Router(config-sr-te-candidate-path-type)# exit
Router(config-sr-te-candidate-path)# exit
```

Cumulative Metric bounds – IGP, TE, hop count, and latency metric bounds are set for the policy. SRTE calculates paths, only when each criterion is satisfied.

```
Router(config-sr-te)# on-demand color 1000 dynamic bounds cumulative
Router(config-sr-te-odc-bounds-type)# type igp 100
Router(config-sr-te-odc-bounds-type)# type te 60
Router(config-sr-te-odc-bounds-type)# type hopcount 6
Router(config-sr-te-odc-bounds-type)# type latency 1000
Router(config-sr-te-odc-bounds-type)# commit
```

To set the maximum number of attempts for computing paths that satisfy the cumulative metric bounds criteria, use the **kshortest-paths** command.

```
Router# configure terminal
Router(config)# segment-routing traffic-eng
Router(config-sr-te)# kshortest-paths 120
Router(config-sr-te)# commit
```

Verification

Use this command to view SR policy configuration details. Pointers:

- The **Number of K-shortest-paths** field displays 4. It means that the K-shortest path algorithm took 4 computations to find the right path. The 4 shortest paths that are computed using K-shortest path algorithm did not respect the cumulative bounds. The fifth shortest path is valid against the bounds.
- The values for the metrics of the actual path (**TE**, **IGP**, **Cumulative Latency** and **Hop count** values in the **Dynamic** section) are within the configured cumulative metric bounds.

```
Router# show segment-routing traffic-eng policy color 2

Color: 2, End-point: 192.168.0.2
  Name: srte_c_2_ep_192.168.0.2
  Status:
    Admin: up  Operational: up for 3d02h (since Dec 15 12:13:21.993)

  Candidate-paths:

    Preference: 100 (configuration) (active)

      Name: fromAtoB_XTC
      Requested BSID: dynamic
      Constraints:
        Protection Type: protected-preferred
        Affinity:
          exclude-any:
          red
        Maximum SID Depth: 10
        IGP Metric Bound: 10
        TE Metric Bound: 60
        Latency Metric Bound: 55
        Hopcount Metric Bound: 4

      Dynamic (valid)

        Metric Type: TE,   Path Accumulated Metric: 52
        Number of K-shortest-paths: 4
        TE Cumulative Metric: 52
        IGP Cumulative Metric: 3
        Cumulative Latency: 52
        Hop count: 3
          16004 [Prefix-SID, 192.168.0.4]
          24003 [Adjacency-SID, 10.16.16.2 - 10.16.16.5]
```

```
        24001 [Adjacency-SID, 10.14.14.5 - 10.14.14.4]

Attributes:

  Binding SID: 24011
  Forward Class: Not Configured
  Steering labeled-services disabled: no
  Steering BGP disabled: no
  IPv6 caps enable: yes
  Invalidation drop enabled: no
```

SR-TE 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).

An **explicit** path is a specified SID-list or set of SID-lists.

An SR-TE policy initiates a single (selected) path in RIB/FIB. This is the preferred valid candidate path. A path is selected when the path is valid and its preference is the best among all candidate paths for that policy.

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

A candidate path has the following characteristics:

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

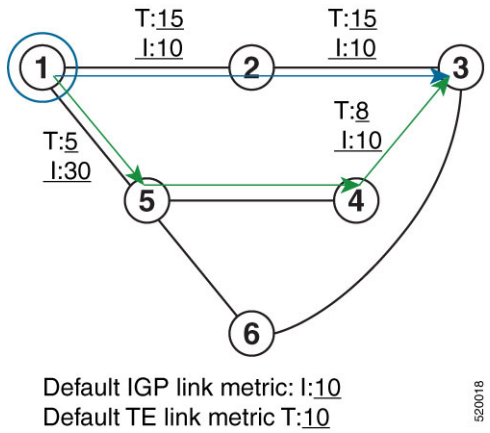
Dynamic Paths

Optimization Objectives

Optimization objectives allow the head-end router to compute a SID-list that expresses the shortest dynamic path according to the selected metric type:

- IGP metric — Refer to the "Implementing IS-IS" and "Implementing OSPF" chapters in the *Routing Configuration Guide for Cisco 8000 Series Routers*.
- TE metric — See the Configure Interface TE Metrics, on page 20 section for information about configuring TE metrics.
- Delay — See the Configure Performance Measurement chapter for information about measuring delay for links or SR policies.

This example shows a dynamic path from head-end router 1 to end-point router 3 that minimizes IGP or TE metric:



- The blue path uses the minimum IGP metric: Min-Metric (1 → 3, IGP) = SID-list <16003>; cumulative IGP metric: 20
- The green path uses the minimum TE metric: Min-Metric (1 → 3, TE) = SID-list <16005, 16004, 16003>; cumulative TE metric: 23

Configure Interface TE Metrics

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

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# interface type interface-path-id
Router(config-sr-te-if)# metric value
```

Configuring TE Metric: Example

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

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

Constraints

Constraints allow the head-end router to compute a dynamic path according to the selected metric type:

- TE 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 21 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 32256 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 19 section.
2. Define the constraints. See the Constraints, on page 20 section.
3. Create the policy.

The following example shows a configuration of an SR policy at an SR-TE head-end router. The policy has a dynamic path with optimization objectives and affinity constraints computed by the head-end router.

```
segment-routing
 traffic-eng
  policy foo
   color 100 end-point ipv4 10.1.1.2
   candidate-paths
    preference 100
     dynamic
      metric
       type te
      !
     !
     constraints
      affinity
```

```
      exclude-any
       name RED
      !
     !
    !
   !
  !
 !
```

The following example shows a configuration of an SR policy at an SR-TE head-end router. The policy has a dynamic path with optimization objectives and affinity constraints computed by the SR-PCE.

```
segment-routing
 traffic-eng
  policy baa
   color 101 end-point ipv4 10.1.1.2
   candidate-paths
    preference 100
     dynamic
      pcep
      !
      metric
       type te
      !
     !
     constraints
      affinity
       exclude-any
        name BLUE
       !
      !
     !
    !
   !
  !
```

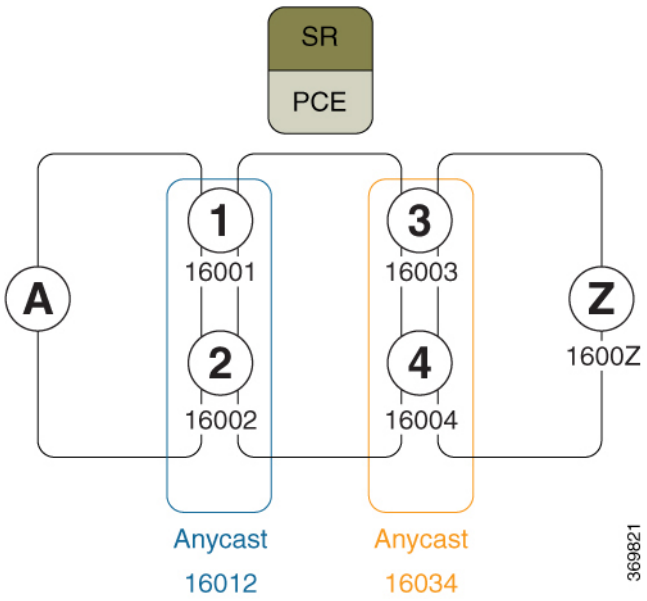
Anycast SID-Aware Path Computation

An Anycast SID is a type of prefix SID that identifies a set of nodes and is configured with n-flag clear. The et 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 and Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface.

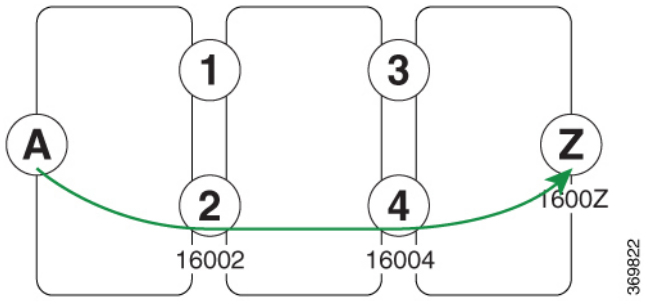
This example shows how Anycast SIDs are inserted into a computed SID list.

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

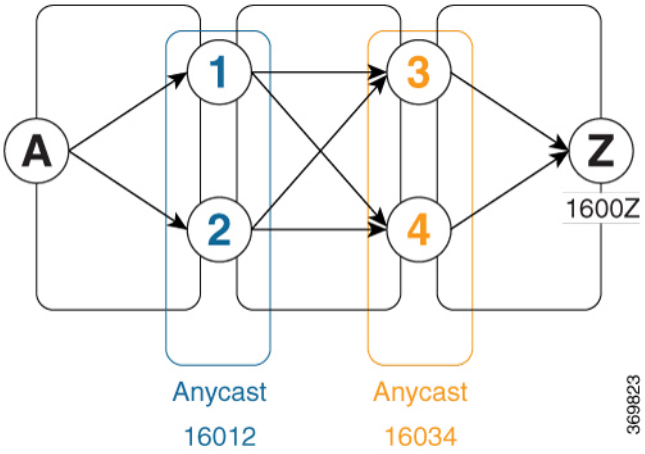
Before considering Anycast SIDs, the head-end router or SR-PCE computes the SID list.



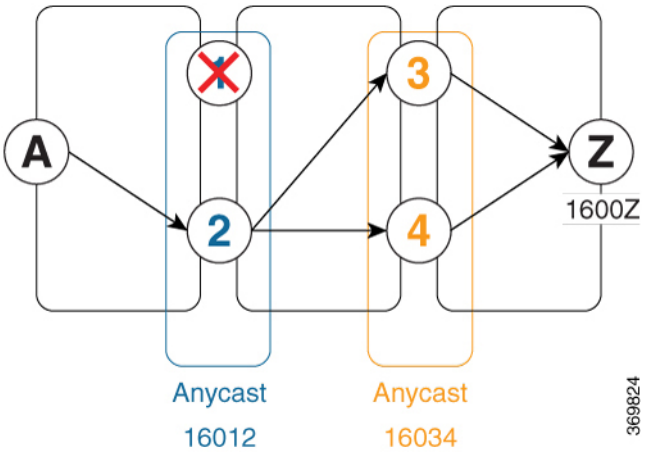
In this case, the optimized computed path from router A to router Z is 16002 > 16004 > 1600Z

The path computation process reiterates the original SID-list and replaces node SIDs with Anycast SIDs (when possible). SR-TE verifies that the Anycast-encoded SID list maintains an optimum path and does not violate any path constraints (link affinity, metric bounds). If the SID list is verified, then the Anycast-encoded SID list is signaled and instantiated in the forwarding.

Using the Anycast-encoded SID list, the optimized computed path from router A to router Z is 16012 > 16034 > 1600Z. The Anycast SID-aware path computation provides load-balancing.



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 router A to router 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 and Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface.

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 include Anycast SIDs into the computed paths of the following 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
```

Explicit Path with Affinity Constraint Validation for Anycast SIDs

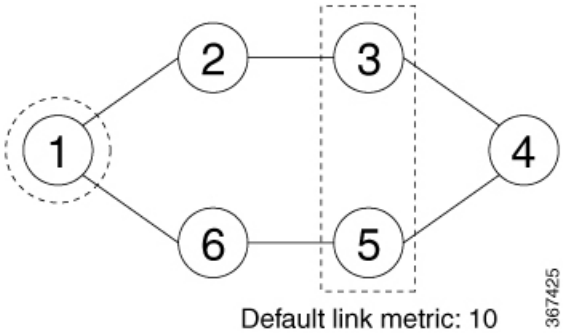
Note For information about configuring Anycast SIDs, see Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface or Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface.

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

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

Affinity Support for Anycast SIDs: Examples

In the following examples, nodes 3 and 5 advertise the same Anycast prefix (10.1.1.8) and assign the same prefix SID (16100).

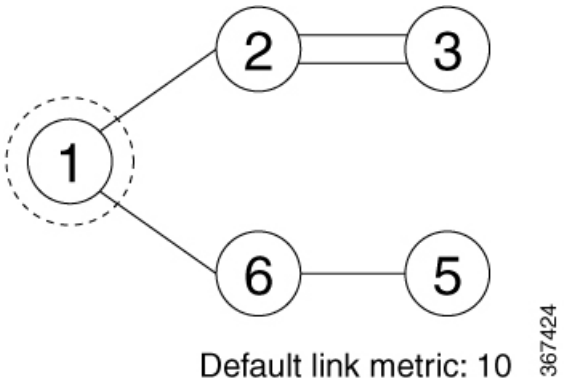


Node 1 uses the following SR-TE policy:

```
segment-routing
 traffic-eng
  policy POLICY1
   color 20 end-point ipv4 10.1.1.4
   binding-sid mpls 1000
   candidate-paths
    preference 100
     explicit segment-list SIDLIST1
     constraints
      affinity
       exclude-any
        red
  segment-list name SIDLIST1
   index 10 address ipv4 192.68.100.100
   index 20 address ipv4 10.4.4.4
```

Affinity Constraint Validation With ECMP Anycast SID: Example

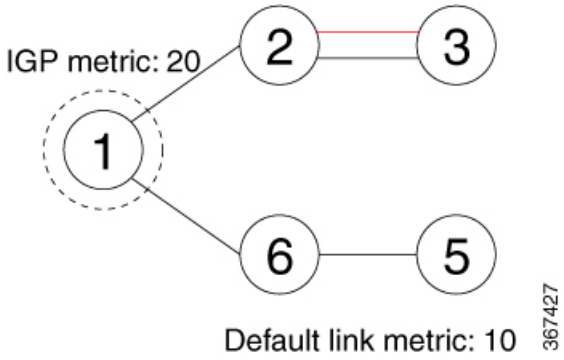
In this example, the shortest path to both node 3 and node 5 has an equal accumulative IGP metric of 20. Both paths are validated against affinity constraints.




```
Name: POLICY1 (Color: 2, End-point: 198.51.100.6)
  Status:
    Admin: up  Operational: up for 00:03:52 (since Jan 24 01:52:14.215)
  Candidate-paths:
    Preference 100:
      Constraints:
        Affinity:
          exclude-any: red
      Explicit: segment-list SIDLIST1 (active)
        Weight: 0, Metric Type: IGP
          16100 [Prefix-SID, 10.1.1.8]
          16004 [Prefix-SID, 10.4.4.4]
```

Affinity Constraint Validation With Non-ECMP Anycast SID: Example

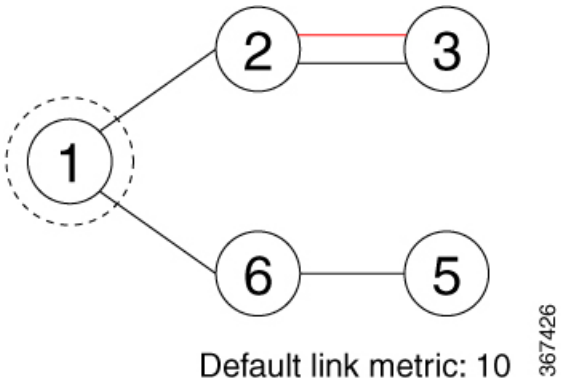
In this example, the shortest path to node 5 has an accumulative IGP metric of 20, and the shortest path to node 3 has an accumulative IGP metric of 30. Only the shortest path to node 5 is validated against affinity constraints.



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

Invalid Path Based on Affinity Constraint: Example

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




```
SR-TE policy database
---------------------
Name: POLICY1 (Color: 2, End-point: 198.51.100.6)
  Status:
    Admin: up  Operational: up for 00:03:52 (since Jan 24 01:52:14.215)
  Candidate-paths:
    Preference 100:
      Constraints:
        Affinity:
          exclude-any: red
      Explicit: segment-list SIDLIST1 (inactive)
      Inactive Reason: Link [10.2.21.23,10.2.21.32] failed to satisfy affinity exclude-any
 constraint=0x00000008, link attributes=0x0000000A
```

Explicit Paths

SR-TE Policy with Explicit Path

Table 3: Feature History Table

Feature Name	Release Information	Feature Description
SR-TE Explicit Segment Lists with Mix of IPv4 and IPv6 Segments	Release 7.9.1	This feature allows you to configure an explicit segment list with IPv4 addresses and include an IPv6 address as a non-first SID. This feature allows you to deploy a centralized BGP EPE solution for 6PE in an SR-MPLS network where the last segment is associated with a EPE-enabled BGPv6 neighbor.

An explicit segment list is defined as a sequence of one or more segments. A segment can be configured as an IP address or an MPLS label representing a node or a link.

An explicit segment list can be configured with the following:

- IP-defined segments

- MPLS label-defined segments
- A combination of IP-defined segments and MPLS label-defined segments

Behaviors and Limitations

- An IP-defined segment can be associated with an IPv4 or IPv6 address (for example, a link or a Loopback address).
- An IPv6 address cannot be the first segment of the segment list.
  - A segment defined with an IPv6 address (for example, IPv6 EPE SID) enables use-cases such as Use Case: Centralized BGP EPE for 6PE in an SR-MPLS Network where the last segment of the explicit segment list is associated with an EPE-enabled BGPv6 neighbor.
- The local MPLS label assigned to a learned BGP prefix SID can be configured as the first segment of a segment list.
- When a segment of the segment list is defined as an MPLS label, subsequent segments can only be configured as MPLS labels.

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 IP addresses:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list name SIDLIST1
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls adjacency 10.1.1.3
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
```

Create 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 IP addresses and MPLS labels:

```
Router(config-sr-te)# segment-list name SIDLIST3
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls label 16003
Router(config-sr-te-sl)# index 30 mpls label 16004
Router(config-sr-te-sl)# exit
```

Create a segment list with IPv4 and IPv6 addresses:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list name SIDLIST4
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls adjacency 10.1.1.3
Router(config-sr-te-sl)# index 30 mpls adjacency 2001:db8:10:1:1::100
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 100
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST2
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# preference 200
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST1
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST4
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
```

Running Configuration

```
Router# show running-configuration
segment-routing
 traffic-eng
  segment-list SIDLIST1
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls adjacency 10.1.1.3
   index 30 mpls adjacency 10.1.1.4
  !
  segment-list SIDLIST2
   index 10 mpls label 16002
   index 20 mpls label 16003
   index 30 mpls label 16004
  !
  segment-list SIDLIST3
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls label 16003
   index 30 mpls label 16004
  !

  segment-list SIDLIST4
   index 10 mpls adjacency 10.1.1.2
   index 10 mpls adjacency 10.1.1.3
   index 10 mpls adjacency 2001:db8:10:1:1::100
  !

  policy POLICY2
   color 20 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     explicit segment-list SIDLIST1
     !
    !
    preference 200
     explicit segment-list SIDLIST2
     !
     explicit segment-list SIDLIST4
```

```
     !
    !
   !
  !
 !
!
```

Verification

This feature provides for displaying detailed segment list information. This is in addition to the current behavior of displaying segment list information from active policies. For active candidate paths, the status of segment list will either be valid or invalid. If the segment list is invalid, the reason for its invalidity along with the entire label/IP stack of segment list is displayed. For inactive candidate paths, the status of segment list will always be inactive. Since the validity of segment list under inactive path is not checked, it is always displayed inactive.

Verify the SR-TE policy configuration using:

```
Router# show segment-routing traffic-eng policy name srte_c_20_ep_10.1.1.4

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

Color: 20, End-point: 10.1.1.4
  Name: srte_c_20_ep_10.1.1.4
  Status:
    Admin: up  Operational: up for 00:00:15 (since Jul 14 00:53:10.615)
  Candidate-paths:
    Preference: 200 (configuration) (active)
      Name: POLICY2
      Requested BSID: dynamic
        Protection Type: protected-preferred
        Maximum SID Depth: 8
      Explicit: segment-list SIDLIST2 (active)
        Weight: 1, Metric Type: TE
          16002
          16003
          16004

    Preference: 100 (configuration) (inactive)
      Name: POLICY2
      Requested BSID: dynamic
        Protection Type: protected-preferred
        Maximum SID Depth: 8
      Explicit: segment-list SIDLIST1 (inactive)
        Weight: 1, Metric Type: TE
          [Adjacency-SID, 10.1.1.2 - <None>]
          [Adjacency-SID, 10.1.1.3 - <None>]
          [Adjacency-SID, 10.1.1.4 - <None>]
    Attributes:
    Binding SID: 51301
   Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
```

Configuring Explicit Path with Affinity Constraint Validation

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

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

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


/* Associate affinity constraints for SR-TE policies
Router(config-sr-te)# segment-list name SIDLIST1
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 20 mpls adjacency 10.2.2.23
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list name SIDLIST2
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.2
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list name SIDLIST3
Router(config-sr-te-sl)# index 10 mpls adjacency 10.1.1.5
Router(config-sr-te-sl)# index 30 mpls adjacency 10.1.1.4
Router(config-sr-te-sl)# exit

Router(config-sr-te)# policy POLICY1
Router(config-sr-te-policy)# color 20 end-point ipv4 10.1.1.4
Router(config-sr-te-policy)# binding-sid mpls 1000
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 200
Router(config-sr-te-policy-path-pref)# constraints affinity exclude-any red
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST1
Router(config-sr-te-pp-info)# exit
```

```
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST2
Router(config-sr-te-pp-info)# exit
Router(config-sr-te-policy-path-pref)# exit
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-policy-path-pref)# explicit segment-list SIDLIST3
```

Running Configuration

```
Router# show running-configuration
segment-routing
 traffic-eng
  interface GigabitEthernet0/0/0/0
   affinity
    name blue
   !
  !
  interface GigabitEthernet0/0/0/1
   affinity
    name blue
    name green
   !
  !
  segment-list SIDLIST1
   index 10 mpls adjacency 10.1.1.2
   index 20 mpls adjacency 10.2.2.23
   index 30 mpls adjacency 10.1.1.4
  !
  segment-list SIDLIST2
   index 10 mpls adjacency 10.1.1.2
   index 30 mpls adjacency 10.1.1.4
  !
  segment-list SIDLIST3
   index 10 mpls adjacency 10.1.1.5
   index 30 mpls adjacency 10.1.1.4
  !
  policy POLICY1
   binding-sid mpls 1000
   color 20 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     explicit segment-list SIDLIST3
     !
    !
    preference 200
     explicit segment-list SIDLIST1
     !
     explicit segment-list SIDLIST2
     !
     constraints
      affinity
       exclude-any
        name red
       !
      !
     !
    !
   !
  !
  affinity-map
   name red bit-position 2
   name blue bit-position 0
   name green bit-position 1
  !
```

```
 !
!
```

Verification

Note Use the auto-generated SR policy name assigned by the router. Auto-generated SR policy names use the following naming convention: **srte_c_***color-value***_ep_***endpoint-address*. For example, **srte_c_20_ep_10.1.1.4**.

```
RP/0/RP0/CPU0:ios# 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: down for 00:07:22 (since Feb 19 17:14:55.564)
  Candidate-paths:
    Preference: 200 (configuration)
      Name: POLICY1
      Requested BSID: 1000
      Constraints:
        Protection Type: protected-preferred
        Affinity:
          exclude-any:
          red
        Maximum SID Depth: 8
      Explicit: segment-list SIDLIST1 (active)
        Weight: 1, Metric Type: TE
      Explicit: segment-list SIDLIST2 (active)
        Weight: 1, Metric Type: TE
    Preference: 100 (configuration)
      Name: POLICY1
      Requested BSID: 1000
      Explicit: segment-list SIDLIST3 (active)
        Weight: 1, Metric Type: TE
  Attributes:
    Forward Class: 0
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: no
    Invalidation drop enabled: no
```

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[password {clear | encrypted} LINE]
Router(config-sr-te-pcc)# pce address ipv4 PCE-address[keychain WORD]
```

Configure PCEP-Related Timers

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

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

Use the **timers deadtimer** command to specify how long the remote peers wait before bringing down the PCEP session if no PCEP messages are received from this PCC. The range is from 1 to 255 seconds; the default value is 120.

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

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

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

PCE-Initiated SR Policy Timers

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

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

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

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

Note Multicast follows the same timer for Point-to-Multipoint (P2MP) policies:

- The default interval is 10 minutes.
- You can set the internal timer to '0' to ensure that forwarding states remain active indefinitely, even if the PCE is down for more than 5 minutes.
- If the PCE is down for more than 5 minutes, it is because the PCE is going down or the reachability to PCE going down.

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 (5).

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

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

- Up to 5 transport labels when no service labels are imposed
- Up to 3 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, where the PCC allows only the PCE with the lowest precedence to initiate policies.

```
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

Procedure

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose
Step 2	**router bgp** *as-number* **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** *ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 10.1.1.1`	Configures the local router with a specified router ID.
Step 4	**address-family** {**ipv4** | **ipv6**} **sr-policy** **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 sr-policy`	Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.
Step 5	**exit**	
Step 6	**neighbor** *ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.0.1`	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.
Step 7	**remote-as** *as-number* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 1`	Creates a neighbor and assigns a remote autonomous system number to it.
Step 8	**address-family** {**ipv4** | **ipv6**} **sr-policy** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 sr-policy`	Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.
Step 9	**route-policy** *route-policy-name* {**in** | **out**} **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass out`	Applies the specified policy to IPv4 or IPv6 unicast routes.

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

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

```
router bgp 65000
bgp router-id 10.1.1.1
 !
 address-family ipv4 sr-policy
 !
 address-family ipv6 sr-policy
 !
 neighbor 10.1.3.1
  remote-as 10
  description *** eBGP session to BGP SRTE controller ***
  address-family ipv4 sr-policy
   route-policy pass in
   route-policy pass out
  !
  address-family ipv6 sr-policy
   route-policy pass in
   route-policy pass out
  !
 !
!
```

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

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

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

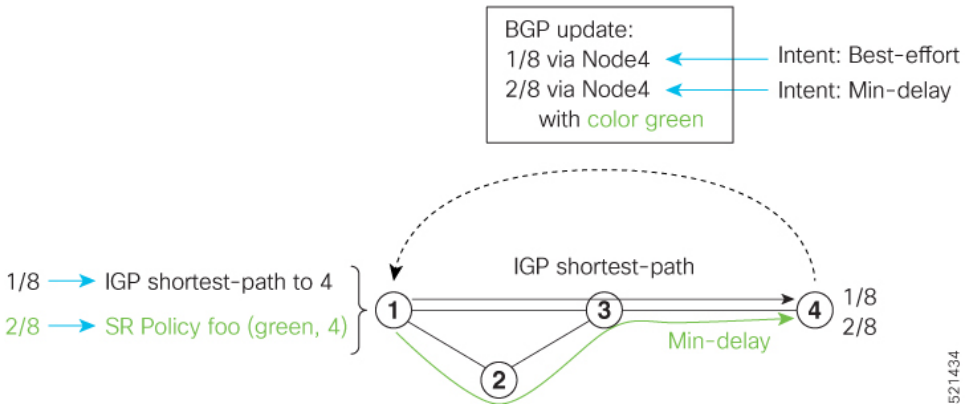
Traffic Steering

Automated Steering

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

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

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



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

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

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

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

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

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 the Color-Only Flag, on page 44.

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 the Color-Only 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 43 for information about color-only steering (NULL end-point).

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

co-flag 00	**1.** The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>. **2.** If a policy does not exist, then IGP path for the next-hop N is chosen.

co-flag 01	**1.** The BGP next-hop and color <N, C> is matched with an SR-TE policy of same <N, C>. **2.** If a policy does not exist, then an SR-TE policy with NULL end-point with the same address-family as N and color C is chosen. **3.** If a policy with NULL end-point with same address-family as N does not exist, then an SR-TE policy with any NULL end-point and color C is chosen. **4.** If no match is found, then IGP path for the next-hop N is chosen.

Configuration Example

```
Router(config)# extcommunity-set opaque overlay-color
Router(config-ext)# 1 co-flag 01
Router(config-ext)# end-set
Router(config)#
Router(config)# route-policy color
Router(config-rpl)# if destination in (10.5.5.1/32) then
Router(config-rpl-if)# set extcommunity color overlay-color
Router(config-rpl-if)# endif
Router(config-rpl)# pass
Router(config-rpl)# end-policy
Router(config)#
```

Address-Family Agnostic Automated Steering

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

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

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

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

Disable IPv6 Encapsulation

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

Per-Flow Automated Steering

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
SR-TE: Per Flow Automated Steering	Release 7.11.1	This feature extends the SR-TE automated steering capabilities with per-flow granularity. Before this feature, per-destination automated steering (AS) dynamically steered all traffic flows destined to a BGP service prefix over a single path of an SR policy. With this feature, Per-Flow AS dynamically steers traffic flows destined to a BGP service prefix over different paths across the network. Traffic flows are determined based on the attributes of incoming packets (for example, source/destination IP address or QoS markings). Per-flow AS is realized using a Per-Flow SR policy (PFP). PFPs offer the user the flexibility to deliver the desired transport paths for different traffic flows.

Per-destination automated steering (AS) dynamically steered all traffic flows destined to a BGP service prefix over a single path of an SR policy. This type of policy is called a Per-Destination Policy (PDP).

The steering of traffic through a Segment Routing (SR) policy is based on the candidate paths of that policy. For a given policy, a candidate path specifies the path to be used to steer traffic to the policy's destination. The policy determines which candidate path to use based on the candidate path's preference and state. The candidate path that is valid and has the highest preference is used to steer all traffic using the given policy. This type of policy is called a Per-Destination Policy (PDP).

Per-Flow Automated Traffic Steering using SR-TE Policies introduces a way to steer traffic on an SR policy based on the attributes of the incoming packets, called a Per-Flow Policy (PFP).

These policies can be used to control and manage the behavior of traffic flows within a network. Forward Class (FC) are groups of flows that share similar characteristics or requirements. These classes are defined based on criteria such as QoS requirements, application types, or other factors. Each flow is assigned to a specific FC based on its attributes, as determined by the Per-Flow Policy. Multiple flows can be part of the same FC. The FC is represented as a numeric value of 0 to 7, providing up to 8 options to the endpoint.

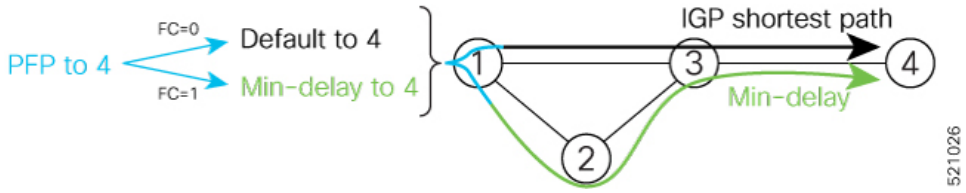
A PFP defines an array of FC-to-PDP mappings. A PFP can then be used to steer traffic into a given PDP based on the FC assigned to a packet.

As with PDPs, PFPs are identified by a {headend, color, endpoint} tuple. The color associated with a given FC corresponds to a valid PDP policy of that color and same endpoint as the parent PFP. So PFP policies contain mappings of different FCs to valid PDP policies of different colors.

Every PFP has an FC designated as its default FC. The default FC is associated to packets with an undefined FC under the PFP, or for packets with an FC that has no valid PDP policy.

The following example shows a per-flow policy from Node1 to Node4:

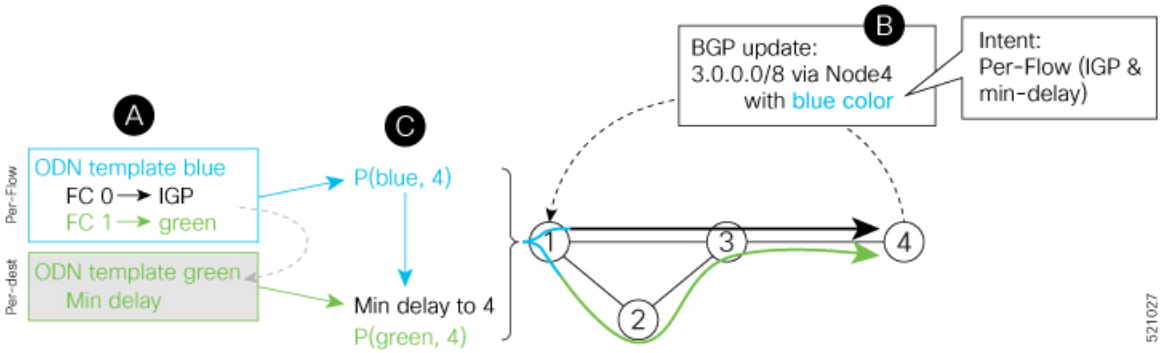
Figure 1: PFP Example



- FC=0 (default) -> shortest path to Node4
  - IGP shortest path = 16004
- FC=1 -> Min-delay path to Node4
  - SID list = {16002,16004}

The same on-demand instantiation behaviors of PDPs apply to PFPs. For example, an edge node automatically (on demand) instantiates Per-Flow SR Policy paths to an endpoint by service route signaling. Automated Steering steers the service route in the matching SR Policy.

Figure 2: PFP with ODN Example



Like PDPs, PFPs have a binding SID (BSID). Existing SR-TE automated steering (AS) mechanisms for labeled traffic (via BSID) and unlabeled traffic (via BGP) onto a PFP is similar to that of a PDP. The classification policy on the ingress interface marks the packet with an FC based on the configured class-map. The packet is then steered to the PDP that corresponds to that FC.

Usage Guidelines and Limitations

The following guidelines and limitations apply to the platform when acting as a head-end of a PFP policy:

- BGP IPv4 unicast over PFP (steered via ODN/AS) is supported
- BGP IPv6 unicast (with IPv4 next-hop [6PE]) over PFP (steered via ODN/AS) is supported

- BGP IPv6 unicast (with IPv6 next-hop) over PFP (steered via ODN/AS) is supported
- BGP VPNv4 over PFP (steered via ODN/AS) is supported
- BGP VPNv6 (6VPE) over PFP (steered via ODN/AS) is supported
- BGP EVPN over PFP is not supported
- Pseudowire and VPLS over PFP are not supported
- BGP multipath is supported
- BGP PIC is not supported
- Labeled traffic (Binding SID as top-most label in the stack) steered over PFP is supported
- When not explicitly configured, FC 0 is the default FC.
- A PFP is considered valid as long as its default FC has a valid PDP.
- An ingress QoS policy applied to an input interface is used to classify flows and set corresponding MPLS experimental values.
- The following counters are supported:
  - PFP's BSID counter (packet, bytes)
  - Per-FC counters (packet, byte)
    - Collected from the PDP's segment-list-per-path egress counters
    - If an SR policy is used for more than one purpose (as a regular policy as well as a PDP under one or more PFPs), then the collected counters will represent the aggregate of all contributions. To preserve independent counters, it is recommended that an SR policy be used only for one purpose.
- Based on the following fields, the inbound packet classification is supported:
  - IP precedence
  - IP DSCP
  - L3 ACL-based (L3 source/destination IP; L4 source/destination port)
  - MPLS EXP
- A color associated with a PFP SR policy cannot be used by a non-PFP SR policy. For example, if a per-flow ODN template for color 100 is configured, then the system will reject the configuration of any non-PFP SR policy using the same color. You must assign different color value ranges for PFP and non-PFP SR policies.

Configuring ODN Template for PFP Policies: Example

The following example depicts an ODN template for PFP policies that includes three FCs.

The example also includes the corresponding ODN templates for PDPs as follows:

- FC0 (default FC) mapped to color 10 = Min IGP path

- FC1 mapped to color 20 = Flex Algo 128 path
- FC2 mapped to color 30 = Flex Algo 129 path

```
segment-routing
 traffic-eng
  on-demand color 10
   dynamic
    metric
     type igp
    !
   !
  !
  on-demand color 20
   constraints
    segments
     sid-algorithm 128
    !
   !
  !
  on-demand color 30
   constraints
    segments
     sid-algorithm 129
    !
   !
  !
  on-demand color 1000
   per-flow
    forward-class 0 color 10
    forward-class 1 color 20
    forward-class 2 color 30
```

Manually Configuring a PFP and PDPs: Example

The following example depicts a manually defined PFP that includes three FCs and corresponding manually defined PDPs.

The example also includes the corresponding PDPs as follows:

- FC0 (default FC) mapped to color 10 = Min IGP path
- FC1 mapped to color 20 = Min TE path
- FC2 mapped to color 30 = Min delay path

```
segment-routing
 traffic-eng
  policy MyPerFlow
   color 1000 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     per-flow
      forward-class 0 color 10
      forward-class 1 color 20
      forward-class 2 color 30
  !
  policy MyLowIGP
   color 10 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
```

```
    dynamic
     metric type igp
  !
  policy MyLowTE
   color 20 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     dynamic
      metric type te
  !
  policy MyLowDelay
   color 30 end-point ipv4 10.1.1.4
   candidate-paths
    preference 100
     dynamic
      metric type delay
```

Configuring Ingress Classification: Example

An ingress classification policy is used to classify and mark traffic to a corresponding forwarding class.

The following shows an example of such ingress classification policy:

```
class-map match-any MinDelay
 match dscp 46
 end-class-map
!
class-map match-any PremiumHosts
 match access-group ipv4 PrioHosts
 end-class-map
 !
!
policy-map MyPerFlowClassificationPolicy
 class MinDelay
  set forward-class 2
 !
 class PremiumHosts
  set forward-class 1
 !
 class class-default
 !
 end-policy-map
!
interface GigabitEthernet0/0/0/0
 description PE_Ingress_Interface
 service-policy input MyPerFlowClassificationPolicy
!
```

Determining Per-Flow Policy State

A PFP is brought down for the following reasons:

- The PDP associated with the default FC is in a down state.
- All FCs are associated with PDPs that are in a down state.
- The FC assigned as the default FC is missing in the forward class mapping.

  Scenario 1—FC 0 (default FC) is not configured in the FC mappings below:

  ```
  policy foo
    color 1 end-point ipv4 10.1.1.1
    per-flow
  ```

```
   forward-class 1 color 10
   forward-class 2 color 20
```

Scenario 2—FC 1 is configured as the default FC, however it is not present in the FC mappings:

```
policy foo
  color 1 end-point ipv4 10.1.1.1
  per-flow
   forward-class 0 color 10
   forward-class 2 color 20
   forward-class default 1
```

Using Binding Segments

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

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

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

BSID can be used in the following cases:

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

Explicit Binding SID

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

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

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

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

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

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

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

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

Stitching SR-TE Polices Using Binding SID: Example

In this example, three SR-TE policies are stitched together to form a seamless end-to-end path from node 1 to node 10. The path is a chain of SR-TE policies stitched together using the binding-SIDs of intermediate policies, providing a seamless end-to-end path.

Figure 3: Stitching SR-TE Polices Using Binding SID

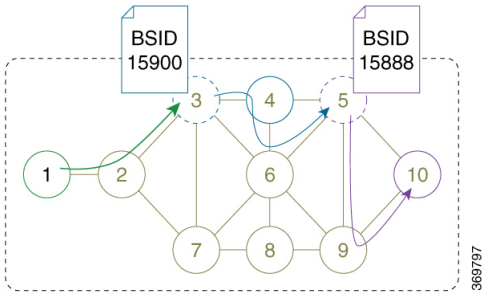


Table 5: Router IP Address

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

Router	Prefix Address	Prefix SID/Adj-SID
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

Procedure

Step 1 On node 5, do the following:

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

b) Define an explicit binding-SID (**mpls label 15888**) allocated from SRLB for the SR-TE policy.

Example:

Node 5

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

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

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

Color: 777, End-point: 10.1.1.10
  Name: srte_c_777_ep_10.1.1.10
  Status:
    Admin: up  Operational: up for 00:00:52 (since Aug 19 07:40:12.662)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: foo
      Requested BSID: 15888
      PCC info:
        Symbolic name: cfg_foo_discr_100
        PLSP-ID: 70
      Explicit: segment-list PATH-9_10 (valid)
        Weight: 1, Metric Type: TE
          16009 [Prefix-SID, 10.1.1.9]
```

```
        16010 [Prefix-SID, 10.1.1.10]
  Attributes:
    Binding SID: 15888 (SRLB)
    Forward Class: 0
    Steering BGP disabled: no
    IPv6 caps enable: yes
```

Step 2 On node 3, do the following:

a) Define an SR-TE policy with an explicit path configured using the following:

- Loopback interface IP address of node 4
- Interface IP address of link between node 4 and node 6
- Loopback interface IP address of node 5
- Binding-SID of the SR-TE policy defined in Step 1 (**mpls label 15888**)

Note This last segment allows the stitching of these policies.

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

Example:

Node 3

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

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

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

Color: 777, End-point: 10.1.1.5
  Name: srte_c_777_ep_10.1.1.5
  Status:
    Admin: up  Operational: up for 00:00:32 (since Aug 19 07:40:32.662)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: baa
      Requested BSID: 15900
      PCC info:
        Symbolic name: cfg_baa_discr_100
        PLSP-ID: 70
```

```
        Explicit: segment-list PATH-4_4-6_5_BSID (valid)
          Weight: 1, Metric Type: TE
            16004 [Prefix-SID, 10.1.1.4]
            80005 [Adjacency-SID, 10.4.6.4 - 10.4.6.6]
            16005 [Prefix-SID, 10.1.1.5]
            15888
    Attributes:
      Binding SID: 15900 (SRLB)
      Forward Class: 0
      Steering BGP disabled: no
      IPv6 caps enable: yes
```

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

Example:

Node 1

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

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

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

Color: 777, End-point: 10.1.1.3
  Name: srte_c_777_ep_10.1.1.3
  Status:
    Admin: up  Operational: up for 00:00:12 (since Aug 19 07:40:52.662)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: bar
      Requested BSID: dynamic
      PCC info:
        Symbolic name: cfg_bar_discr_100
        PLSP-ID: 70
      Explicit: segment-list PATH-3_BSID (valid)
        Weight: 1, Metric Type: TE
          16003 [Prefix-SID, 10.1.1.3]
          15900
  Attributes:
    Binding SID: 80021
    Forward Class: 0
```

```
Steering BGP disabled: no
IPv6 caps enable: yes
```

Static Route Traffic-Steering using SR-TE Policy

Table 6: Feature History Table

Feature Name	Release Information	Feature Description
Static Route Traffic-Steering using SR-TE Policy	Release 7.10.1	IPv4 and IPv6 static routes now leverage the SR policies to aid Segment Routing Traffic Engineering (SR-TE). This facilitates traffic steering because you can now configure IP Static Route with SR static policy.

The Static Route Traffic-Steering using SR-TE Policy feature allows you to specify a Segment Routing (SR) policy when configuring static routes.

For information on configuring static routes, see the "Implementing Static Routes" chapter in the *Routing Configuration Guide for Cisco 8000 Series Routers*.

Configuration Example

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

//configure administrative distance
Router (config-static-afi)# 192.2.0.0/24 sr-policy sample-policy 110

//Configure load metric
Router (config-static-afi)# 192.2.0.0/24 sr-policy sample-policy metric 50

//Install the route in RIB regardless of reachability
Router (config-static-afi)# 192.2.0.0/24 sr-policy sample-policy permanent
```

Running Configuration

```
Router# show running-configuration
 router static
  address-family ipv4 unicast
  192.2.0.1/24 sr-policy sample-policy 110
  192.2.0.1/24 sr-policy sample-policy metric 50
  192.2.0.1/24 sr-policy sample-policy permanent
  !
 !
!
```

Verification Steps

```
Router# show route 192.2.0.0/24

Routing entry for 192.2.0.0/24
```

```
Known via "static", distance 1, metric 0
Installed Jul 28 09:18:25.639 for 00:00:09
Routing Descriptor Blocks
directly connected, via srte_c_2_ep_10.3.3.10, permanent
Route distance is 110, metric is 0, Wt is 50
```

Label Distribution Protocol (LDP) over Segment Routing Traffic Engineering (SR-TE) Policy

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
Label Distribution Protocol over Segment Routing Policy	Release 7.10.1	This feature extends the existing MPLS Label Distribution Protocol (LDP) address family neighbor configuration to specify an SR policy as the targeted end-point, thus integrating LDP over SR-TE in core networks and enabling interoperability between LDP and SR-capable devices. The feature introduces the **neighbor sr-policy** command.

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*.

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.

Note You can use the **show segment-routing traffic-eng policy** command to display the auto generated SR policy name. Auto-generated SR policy name uses 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. Ensure you use the same method for the peer router, it must have similar configurations:

- 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: 1.1.1.2
  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       10.2.2.99       10.1.2.2       10.2.3.2
      10.2.4.2       10.2.22.2      10.2.222.2     10.30.110.132
      10.2.9.2
    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)
```

Autoroute Include

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
Support for mixed SR-TE Policy Autoroute paths, unprotected native paths, and protected (LFA/TI-LFA) native paths	Release 7.11.1	When an SR-TE policy is autoroute announced, an IGP route can use it as one of its nexthops, as well as other native paths for load balancing. If protection is configured under IGP, some of these native paths may have LFA or TI-LFA backup paths. In earlier releases, this mix of SR-TE policy paths and protected native paths wasn't supported. This feature adds support for protected (LFA/TI-LFA) native paths. This functionality is enabled by default.

Feature Name	Release Information	Feature Description
IPv6 'Autoroute Include' Support for an SR-TE Policy with an IPv4 Endpoint	Release 7.5.4	You can configure an SR-TE policy to automatically steer incoming unlabeled IPv6 traffic at the headend router into that SR-TE policy with an IPv4 endpoint. Compared to MPLS/RSVP-TE, SR-TE provides more granular and automated steering techniques. In earlier releases, you could automatically steer only incoming IPv4 traffic for specific or all prefixes. New command: **autoroute include ipv6 all**

You can configure SR-TE policies with Autoroute Include to steer all prefixes and specifically IGP (IS-IS, OSPF) prefixes over non-shortest paths and divert traffic for those prefixes onto the SR-TE policy.

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

The **autoroute include ipv4** {**all** | *address*} option applies Autoroute Destination functionality for all eligible or specified IPv4 prefixes. The *address* option is supported for IS-IS only; it is not supported for OSPF.

The **autoroute include ipv6 all** option applies Autoroute Destination functionality for all eligible IPv6 prefixes.

Usage Guidelines and Limitations

- Autoroute Include for IPv6 is supported for unlabeled IGP prefixes and BGP penultimate next-hop (PNH).
- Autoroute Include supports three metric types:
  - Default (no metric): The path over the SR-TE policy inherits the shortest path metric.
  - Absolute (constant) metric: The shortest path metric to the policy endpoint is replaced with the configured absolute metric. The metric to any prefix that is Autoroute Included is modified to the absolute metric. Use the **autoroute metric constant** *constant-metric* command, where *constant-metric* is from 1 to 2147483647.
  - Relative metric: The shortest path metric to the policy endpoint is modified with the relative value configured (plus or minus). Use the **autoroute metric relative** *relative-metric* command, where *relative-metric* is from -10 to +10.

Note To prevent load-balancing over IGP paths, you can set a metric that is lower than the value considered by the IGP for autorouted destinations. For example, you can use a relative metric of **-1**. By doing this, you can influence the IGP to prefer other paths over the autorouted destinations, effectively preventing load-balancing over those paths.

- The ECMP path-set of an IGP route with a mix of SR-TE Policy paths (Autoroute) and unprotected native paths is supported.
- The ECMP path-set of an IGP route with a mix of SR-TE Policy paths (Autoroute) and protected (LFA/TI-LFA) native paths is supported.

Note Mixed paths are not supported when the following features are enabled:

- Increase in RSVP-TE Tunnel Scale for LDP over TE (using the **hw-module profile cef te-tunnel highscale-ldp-over-te-no-sr-over-srte -** command)
- LDP Over RSVP LSR (using the **hw-module profile cef te-tunnel highscale-no-ldp-over-te** command)
- SR-TE with Next-Hop Independent Scaling Optimization (using the **segment-routing traffic-eng separate-next-hop** command)

- LDP to SR-TE interworking is not supported.

Configuration Examples

The following example shows how to configure autoroute include for all IPv4 prefixes:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)#policy P1
Router(config-sr-te-policy)# color 20 end-point ipv4 10.1.1.2
Router(config-sr-te-policy)# autoroute include ipv4 all
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-pp-index)# explicit segment-list Plist-1
```

The following example shows how to configure autoroute include for the specified IPv4 prefixes:

Note This option is supported for IS-IS only; it is not supported for OSPF.

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# traffic-eng
```

```
Router(config-sr-te)#policy P1
Router(config-sr-te-policy)# color 20 end-point ipv4 10.1.1.2
Router(config-sr-te-policy)# autoroute include ipv4 10.1.1.21/32
Router(config-sr-te-policy)# autoroute include ipv4 10.1.1.23/32
Router(config-sr-te-policy)# autoroute metric constant 1
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 100
Router(config-sr-te-pp-index)# explicit segment-list slist-1
```

The following example shows how to configure the IPv6 autoroute function for an SR-TE policy with an IPv4 endpoint:

```
Router# configure
Router(config)# segment-routing traffic-eng policy pol12
Router(config-sr-te-policy)# autoroute include ipv6 all
Router(config-sr-te-policy)# commit
```

The following example shows how to configure the IPv6 autoroute function for a PCE-instantiated SR-TE policy with an IPv4 endpoint:

```
Router# configure
Router(config)# segment-routing traffic-eng pcc profile 10
Router(config-pcc-prof)# autoroute include ipv6 all
Router(config-pcc-prof)# commit
```

Verification

```
Router# show segment-routing traffic-eng policy name srte_c_20_ep_10.1.1.2 private

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

Color: 20, End-point: 10.1.1.2 ID: 1
  Name: srte_c_20_ep_10.1.1.2
  Status:
    Admin: up  Operational: down for 00:05:57 (since Mar 13 18:08:26.690)
  Candidate-paths:
    Preference: 100 (configuration) (inactive)
      Originator: ASN 0 node-address <None> discriminator: 100
      Name: policy1
      Requested BSID: 15001
        Protection Type: protected-preferred
        Maximum SID Depth: 8
      ID: 1
      Source: <None>
      Stale: no
      Checkpoint flags: 0x00000000
      Autoroute:
        Force SR include: no
        Include IPv6 all: yes
        Prefix: 0.0.0.0/0
      Explicit: segment-list slist1 (inactive)
        Weight: 2, Metric Type: TE
        IGP area: 0

. . .

Router# show isis ipv6 route 2001:131:0:63::1/64 detail

L2 2001:131:0:63::1/64 (41/115) Label: None, low priority
   Installed Feb 22 23:03:14.620 for 00:00:02
     via ::, srte/c/1/ep/10.1.1.2, Label: Exp-Null-v6, SR-TB5-R2, SRGB Base: 21000, Weight:
 0
```

```
     src 0010.9400.0006.00-02, 2002::6702:102


Router# show route ipv6 2001:131:0:63::1/64 detail

Routing entry for 2001:131:0:63::/64
  Known via "isis core-sr", distance 115, metric 41, type level-2
  Installed Feb 22 23:03:14.624 for 00:04:20
  Routing Descriptor Blocks
    directly connected, via srte c_1_ep_10.1.1.2
      Nexthop in Vrf: "default", Table: "default", IPv4 Unicast, Table Id:Oxe0000000
      Route metric 18 41
      Label: OX2 (2)
      Tunnel ID: None
      Binding Label: 0x272e (15001)
      Extended communities count: 0
      Path id:1
      Path ref count: 0
      NHID: 0x0 (Ref: 0)
      MPLS eid:Ox118aa00000002

. . .
```

The following output shows an ECMP path-set of an IGP route with a mix of SR-TE Policy paths (Autoroute Include) and protected (LFA/TI-LFA) native paths:

```
RP/0/RP0/CPU0:R1# show route ipv4 131.0.2.1

Routing entry for 131.0.2.1/32
  Known via "ospf core-sr", distance 110, metric 14, labeled SR(SRMS), type intra area
  Installed Jun  1 21:01:02.789 for 2d00h
  Routing Descriptor Blocks
    101.1.3.2, from 103.2.1.2, via Bundle-Ether1301, Protected
      Route metric is 14
    101.1.5.2, from 103.2.1.2, via Bundle-Ether1501, Backup (Local-LFA)
      Route metric is 15
    100.2.1.1, from 103.2.1.2, via srte_c_700_ep_100.2.1.1
      Route metric is 14
  No advertising protos.
```

SR-TE Automated Steering Without BGP Prefix Path Label

Table 9: Feature History Table

Feature Name	Release Information	Feature Description
SR-TE Automated Steering Without BGP Prefix Path Label	Release 7.9.1	This feature allows traffic to a BGP service route to be steered over an SR-TE policy using automated-steering principles without imposing the service route's prefix label. This feature allows you to deploy a centralized BGP EPE solution for 6PE in an SR-MPLS network. This feature introduces the **bgp prefix-path-label ignore** command.

This feature allows traffic to a BGP service route to be steered over an SR-TE policy using automated-steering principles without imposing the service route's prefix label (see Automated Steering, on page 42). BGP ignores the programming of the label associated with a prefix path (for example, 6PE/VPN label) when recursing onto the BSID of an SR-TE policy with this feature enabled.

This feature allows you to deploy a Use Case: Centralized BGP EPE for 6PE in an SR-MPLS Network.

Usage Guidelines and Limitations

This functionality applies to local/manually configured SR-TE candidate-paths.

This functionality does not apply to on-demand SR-TE candidate-paths triggered by ODN.

This functionality does not apply to SR-TE candidate-paths instantiated via PCEP (PCE-initiated) or BGP-TE.

Configuration

Use the **bgp prefix-path-label ignore** command in SR-TE policy steering config mode to indicate BGP ignores the programming of the label associated with a prefix path (for example, 6PE/VPN label) when recursing onto the BSID of an SR-TE policy with this feature enabled.

```
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# policy FOO
Router(config-sr-te-policy)# steering
Router(config-sr-te-policy-steering)# bgp prefix-path-label ignore
Router(config-sr-te-policy-steering)# exit
Router(config-sr-te-policy)# color 100 end-point ipv4 0.0.0.0
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-sl
```

Verification

The following output displays the SR-TE policy (SR policy color 100, IPv4 null end-point) details showing the ignore prefix label steering behavior:

```
Router# show segment-routing traffic-eng policy candidate-path name FOO private

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

Color: 100, End-point: 0.0.0.0 ID: 3
  Name: srte_c_100_ep_0.0.0.0
  Status:
    Admin: up  Operational: up for 00:10:07 (since Feb  2 12:58:43.554)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Originator: ASN 0 node-address <None> discriminator: 100
      Name: FOO
      Requested BSID: dynamic
      Constraints:
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      ID: 1
      Source: 20.1.0.100
      Stale: no
      Checkpoint flags: 0x00000000
      Steering:
        Client: BGP
          Disabled: no
          Ignore prefix label: yes
      Explicit: segment-list sample-sl (valid)
        Weight: 1, Metric Type: TE
        IGP area: 2
          SID[0]: 16102 [Prefix-SID: 20.1.0.102, Algorithm: 0]
          SID[1]: 16103 [Prefix-SID: 20.1.0.103, Algorithm: 0]
          SID[2]: 24008 [Adjacency-SID, 15:15:15::4 - 15:15:15::5]
  LSPs:

. . .

  Attributes:
    Binding SID: 24030
    Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
    Max Install Standby Candidate Paths: 0
  Notification to clients:
    Binding SID: 24030
    Bandwidth : 0 Kbps (0 Kbps)
    State: UP
    Flags: [add] [ipv6_caps] [ignore_prefix_label]
    Metric Type: NONE
    Metric Value: 2147483647
    Admin Distance: 100
  ifhandle: 0x00000170
  Source: 20.1.0.100
  Transition count: 1
  LSPs created count: 1
  Reoptimizations completed count: 1
  Retry Action Flags: 0x00000000, ()
  Last Retry Timestamp: never (0 seconds ago)
  Policy reference: 0x1f81e50
```

The following output shows that BGP received the ignore prefix label steering behavior for an SR policy color 100 and IPv4 null end-point:

```
Router# show bgp nexthops 0.0.0.0 color 100 | include "BGP prefix label"

    BGP prefix label: [No]
```

The following output shows the details for a IPv6 BGP global route (151:1::/64) learned from an IPv4 next-hop (6PE) that is steered over an SR policy (BSID 24030). BGP programs the prefix path ignoring its label.

```
Router# show bgp ipv6 labeled-unicast 151:1::/64 detail

BGP routing table entry for 151:1::/64
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker               2003         2003
    Local Label: 81718 (no rewrite);
    Flags: 0x003e1001+0x30010000;
Last Modified: Nov 23 16:59:17.891 for 00:00:03
Paths: (400 available, best #1)
  Advertised IPv6 Unicast paths to update-groups (with more than one peer):
    0.2
  Advertised IPv6 Labeled-unicast paths to update-groups (with more than one peer):
    0.3
  Path #1: Received by speaker 0
  Flags: 0xa480000001060205+0x01, import: 0x020
  Advertised IPv6 Unicast paths to update-groups (with more than one peer):
    0.2
  Advertised IPv6 Labeled-unicast paths to update-groups (with more than one peer):
    0.3
  300, (Received from a RR-client)
    5.5.3.1 C:100 (bsid:24030) (admin 100) (metric 2147483647) from 4.4.4.1 (5.5.5.5),
if-handle 0x00000170
      Prefix Label not imposed due to SR policy config
```

Use Case: Centralized BGP EPE for 6PE in an SR-MPLS Network

In this use case, an operator wants to control the egress peering router/egress transit autonomous system (AS) used by selected Internet IPv6 prefixes. To achieve this, SR policies with explicit paths are used to steer traffic to an intended egress peering router and intended egress transit AS. BGP-EPE SIDs are used in order to force traffic onto an intended egress transit AS. Traffic steering follows SR-TE automated-steering principles.

The following key features enable the use-case:

- SR-TE Policy with Explicit Path — Allows the last segment of an SR policy's SID list to be associated with an EPE-enabled BGPv6 neighbor.
- SR-TE Automated Steering Without BGP Prefix Path Label — Allows traffic to an Internet IPv6 prefix to be steered over an SR-TE policy without imposing the 6PE label learned from the original advertising router.

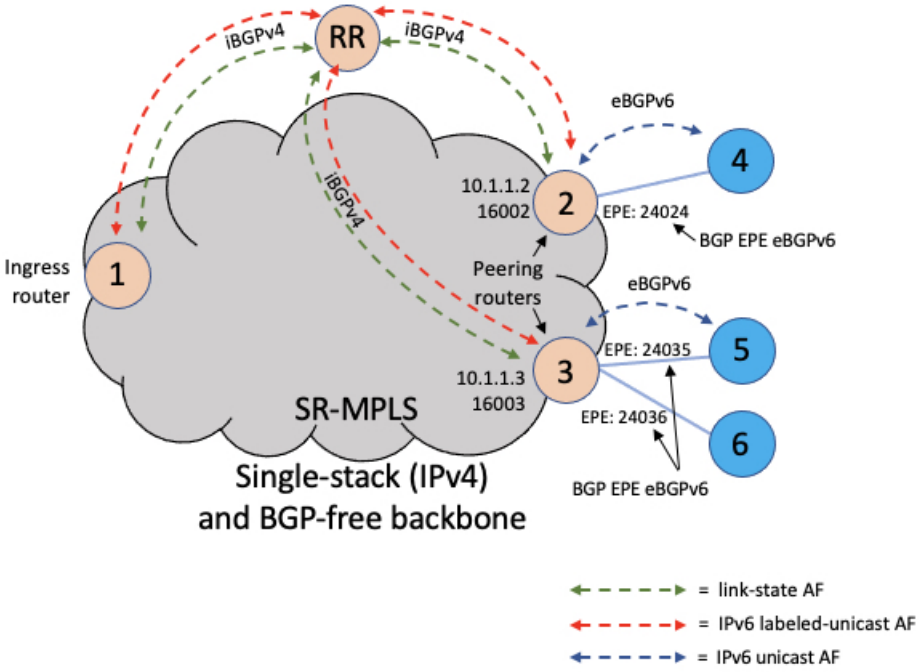
Topology

The below topology shows a single-stack IPv4 SR-MPLS and BGP-free network that delivers Internet IPv6 connectivity using 6PE.

Peering routers 2 and 3 learn IPv6 reachability through transit AS's (ASBR routers 4, 5, 6) via eBGPv6 neighbors.

BGP EPE SIDs are enabled on external BGPv6 neighbors at router 2 (for example, EPE label 24024) and router 3 (for example, EPE labels 24035 and 24036).

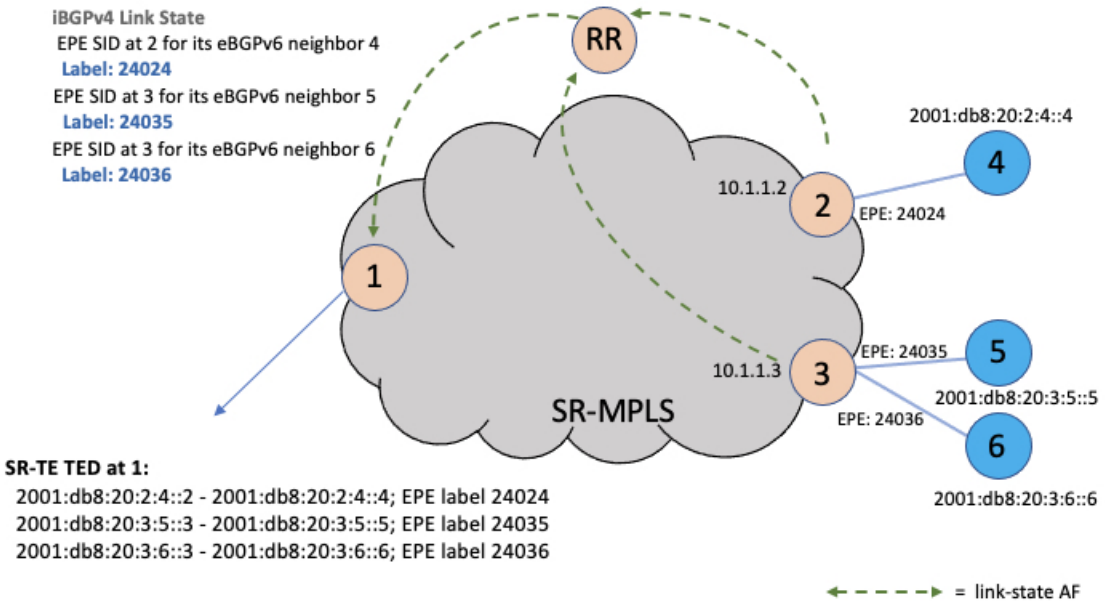
Figure 4: Network Setup



BGP EPE Propagation via BGP-LS

Peering routers 2 and 3 advertise their EPE-enabled neighbors via BGP-LS. As a result, ingress router node 1 learns those EPE-enabled neighbors via BGP-LS. This allows the SR-TE database at the ingress router to include the external links.

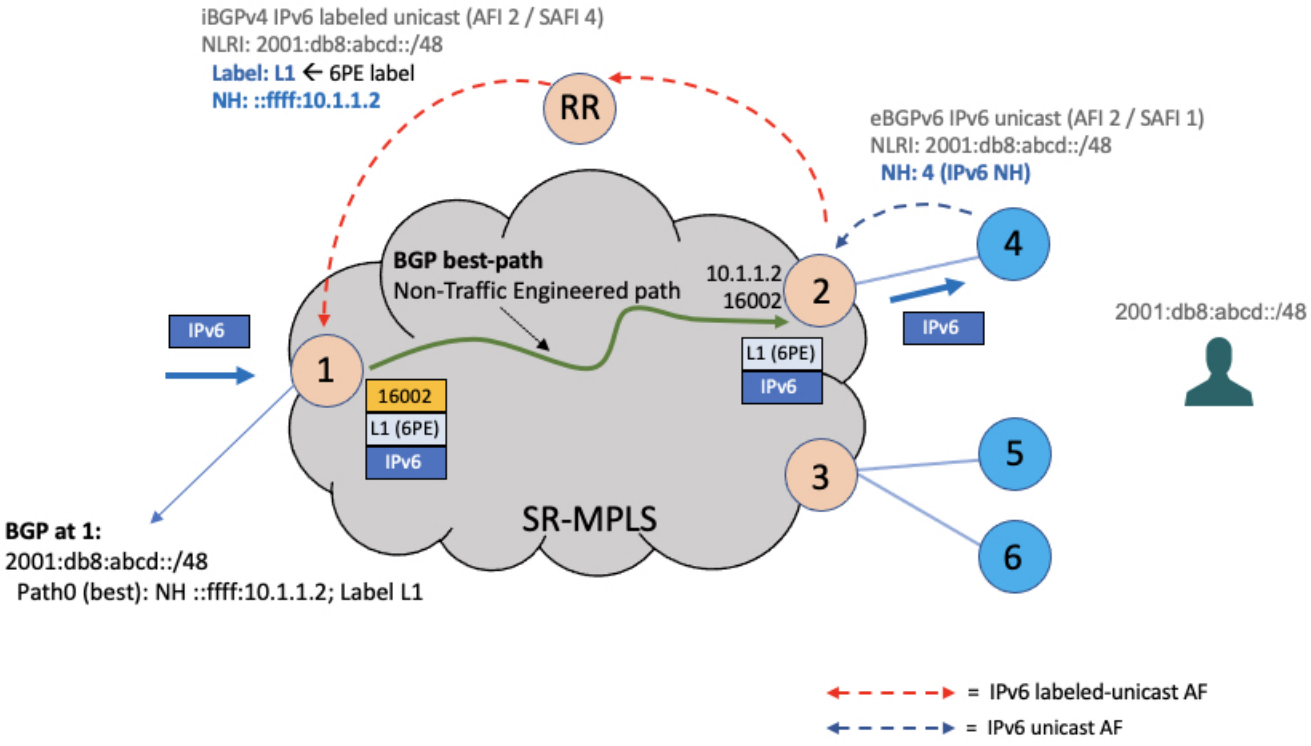
Figure 5: BGP-LS



Steady State (Non-Traffic-Engineered)

At steady state, router 1 selects (as BGP best-path) the path from router 2 for IPv6 prefix 2001:db8:abcd::/48. Traffic to this prefix is sent over the SR-native LSP associated with router 2 (prefix SID 16002) along side the advertised 6PE label.

Figure 6: Steady State



EPE Traffic-Engineered Path

To create a traffic-engineered path that steers traffic to an intended egress peering router/egress transit AS (for example, node 3/ASBR node 6), an SR policy can be configured at ingress border router 1 with the following:

- An IPv4 null (0.0.0.0) end-point, in order to perform color-only automated steering (see Color-Only Automated Steering, on page 43).
- An explicit segment list with SIDs corresponding to the intended egress node (for example, node 3) and the intended egress peering link (for example, ASBR node 6).
- The **bgp prefix-label ignore** steering command in order to indicate BGP to ignore the programming of the 6PE label associated with the prefix path.

```
segment-routing
 traffic-eng
  segment-list sl-to_3-epe_36
   index 10 mpls adjacency 10.1.1.3
   index 20 mpls adjacency 2001:db8:20:3:6::3
  !
  policy FOO
   steering
    bgp prefix-path-label ignore
   !
   color 10 end-point ipv4 0.0.0.0
   candidate-paths
    preference 100
```

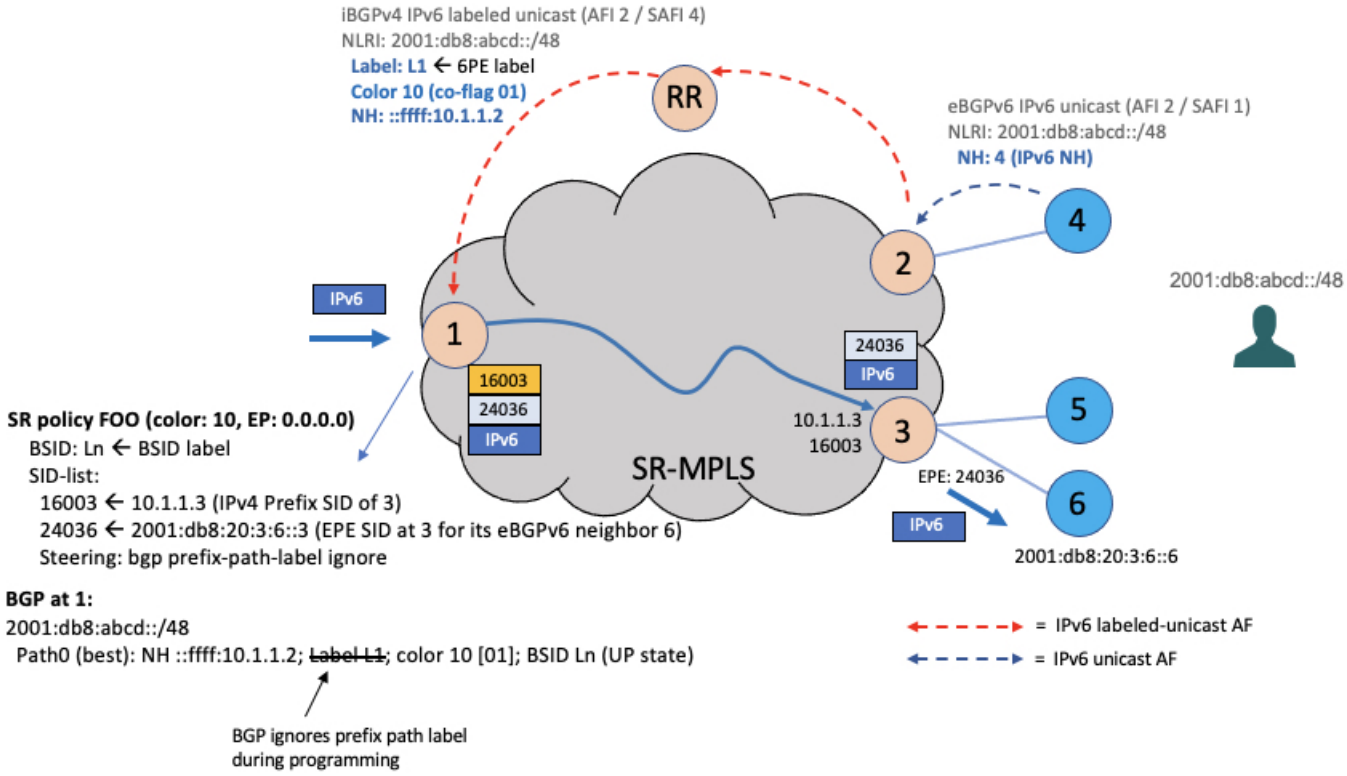
```
explicit segment-list sl-to_3-epe_36
```

When a given IPv6 Internet destination needs to be steered over an intended egress peering router/egress AS, the operator can perform one of the following:

- Advertise a new BGP prefix path from a Route Server that includes a color extended community value equal to the color of the SR-TE policy for the intended egress peering router/egress AS, or
- Apply a color extended community value equal to the color of the SR-TE policy for the intended egress peering router/egress AS at the peering router advertising the best path (for example, node 2), as shown below.

Note The BGP color includes the color-only flag value of 01 in order to allow for color-only automated steering.

Figure 7: EPE Traffic-Engineered Path



The following output depicts the details of the SR-TE policy programmed at the ingress border router node 1 used to send traffic to the egress peering router node 3 and egress AS behind ASBR node 6:

```
Router1# show segment-routing traffic-eng policy candidate-path name FOO private

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

Color: 10, End-point: 0.0.0.0 ID: 3
```

```
  Name: srte_c_10_ep_0.0.0.0
  Status:
    Admin: up  Operational: up for 00:10:07 (since Feb  2 12:58:43.554)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Originator: ASN 0 node-address <None> discriminator: 100
      Name: FOO
      Requested BSID: dynamic
      Constraints:
        Protection Type: protected-preferred
        Maximum SID Depth: 10
      ID: 1
      Source: 10.1.1.1
      Stale: no
      Checkpoint flags: 0x00000000
      Steering:
        Client: BGP
          Disabled: no
          Ignore prefix label: yes
      Explicit: segment-list sl-to_3-epe_36 (valid)
        Weight: 1, Metric Type: TE
        IGP area: 2
          SID[0]: 16003 [Prefix-SID: 10.1.1.3, Algorithm: 0]
          SID[1]: 24036 [Adjacency-SID, 2001:db8:20:3:6::3 - 2001:db8:20:3:6::6]
  LSPs:

. . .

  Attributes:
    Binding SID: 24030
    Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
    Max Install Standby Candidate Paths: 0
  Notification to clients:
    Binding SID: 24030
    Bandwidth : 0 Kbps (0 Kbps)
    State: UP
    Flags: [add] [ipv6_caps] [ignore_prefix_label]
    Metric Type: NONE
    Metric Value: 2147483647
    Admin Distance: 100
  ifhandle: 0x00000170
  Source: 10.1.1.1
  Transition count: 1
  LSPs created count: 1
  Reoptimizations completed count: 1
  Retry Action Flags: 0x00000000, ()
  Last Retry Timestamp: never (0 seconds ago)
  Policy reference: 0x1f81e50
```

The following output depicts the details of the IPv6 BGP global route (2001:db8:abcd::/48) being steered over the binding SID of the previoulsy shown SR-TE policy (24030):

```
Router1# show bgp ipv6 labeled-unicast 2001:db8:abcd::/48 detail

BGP routing table entry for 2001:db8:abcd::/48
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                2003         2003
    Local Label: 81718 (no rewrite);
```

```
    Flags: 0x003e1001+0x30010000;
Last Modified: Nov 23 16:59:17.891 for 00:00:03
Paths: (1 available, best #1)
  Advertised IPv6 Unicast paths to update-groups (with more than one peer):
    0.2
  Advertised IPv6 Labeled-unicast paths to update-groups (with more than one peer):
    0.3
  Path #1: Received by speaker 0
  Flags: 0xa480000001060205+0x01, import: 0x020
  Advertised IPv6 Unicast paths to update-groups (with more than one peer):
    0.2
  Advertised IPv6 Labeled-unicast paths to update-groups (with more than one peer):
    0.3
  300, (Received from a RR-client)
    10.1.1.2 C:10 (bsid:24030) (admin 100) (metric 2147483647) from 10.1.1.100 (10.1.1.2),
 if-handle 0x00000170
      Prefix Label not imposed due to SR policy config
```

Enabling SR-TE with Next-Hop Independent Scaling Optimization

The Next-Hop Independent Scaling Optimization programs an SR-TE policy using a recursive forwarding chain with 2 levels of ECMP. Decoupling next-hop programming results in lower consumption of ASIC resources.

This optimization is disabled by default. Perform the following steps to enable the next-hop independent scaling optimization .

1. Delete any existing SR policies using no form of the command.
2. Enable the optimization using the **segment-routing traffic-eng separate-next-hop** command.

   ```
   Router(config)# segment-routing traffic-eng separate-next-hop
   ```
3. Reload the router.
4. Configure the SR policies again. See Instantiation of an SR Policy, on page 4.

Usage Guidelines and Limitations for Next-Hop Independent Scaling Optimization

The following features are supported:

- SR-TE On-Demand Next Hop/Automated Steering (ODN/AS) for IPv4 BGP and IPv6 BGP global routes
- SR-TE BSID-based AS
- SR-TE head-end path computation
- LFA at SR-TE head-end
- Per-SR policy BSID label counters
- Per-SR policy aggregate counters

- Per-SR policy, per-segment list aggregate counters
- Per-SR policy, per-segment list, per-protocol aggregate counters:
  - Unlabeled IP – Unlabeled IPv4 and IPv6 traffic steered over SR policy
  - Labeled MPLS – Labeled traffic with BSID as top of label stack steered over SR policy

The following features are not supported:

- PCEP at SR-TE head-end
- SR-TE ODN/AS for 6PE, VPNv4, VPNv6 (6vPE), EVPN
- TI-LFA at SR-TE head-end
- SR-TE per-flow policy (PFP)
- SR-TE policy with autoroute-include-based steering
- Static Route Traffic-Steering using SR-TE Policy
- LDP over SR-TE Policy
- Per-SR policy, per-segment list, per-path aggregate counters

Miscellaneous

Segment Routing Encapsulation Object Optimization

Table 10: Feature History Table

Feature Name	Release Information	Feature Description
Segment Routing Encapsulation Object Optimization	Release 7.5.4	The SR Encapsulation object optimization minimizes the forwarding ASIC's Encapsulation resource consumption during programming of an SR-MPLS network, thanks to globally significant label values. With this feature, the forwarding chain of a labeled prefix with ECMP consumes only a single global encapsulation entry in the hardware, instead of an encapsulation entry for each outgoing path. New command: • **hw-module profile cef sropt enable**

When programming an SR-MPLS network, the Segment Routing Encapsulation (Encap) object optimization minimizes the encapsulation resource consumption of the forwarding ASIC. This is because Segment Routing uses globally significant label values.

With this feature, instead of consuming an encapsulation entry for each outgoing path, the forwarding chain of a labeled prefix with ECMP consumes only a single global encapsulation entry.

Usage Guidelines and Limitations

- SR Encap object optimization is triggered only when all ECMP paths of a labeled prefix (primary and backup) perform the same egress action (either all pop or all swap); and have the same outgoing label for the swap egress action. If this condition is not met, then the prefix is programmed with a dedicated Encap object per outgoing path.
- SR Encap object optimization is supported for both labeled IPv4 /32 (SR-MPLSv4) and labeled IPv6 /128 (SR-MPLSv6).
- All paths associated with the prefix (primary and backup) must have the same outgoing label value for SR Encap object optimization to be triggered. For example:
  - For prefixes with LFA backup paths, the SR Encap object optimization is triggered because these backup paths do not require an extra label to be pushed.
  - For prefixes with TI-LFA backup paths requiring extra labels to be pushed, the SR Encap object optimization is not triggered because all the paths associated with the prefix do not have the same outgoing label value.
- Per-label per-interface egress counters are not supported when SR Encap object optimization is enabled. Instead, per-label aggregate egress counters are supported.
- SR MicroLoop Avoidance is not supported when SR Encap object optimization is enabled.

Configuration

Use the **hw-module profile cef sropt enable** command to enable SR Encap object optimization.

Note After you enter this command, you must reload the router.

```
Router(config)# hw-module profile cef sropt enable

In order to activate/deactivate SROPT feature, you must manually reload the chassis/all
line cards

Router(config)# commit

Router(config)# end

Router# show hw-module profile cef

-----------------------------------------------------------------
Knob                          Status          Applied   Action
-----------------------------------------------------------------
CBF Enable                    Unconfigured    N/A       None
CBF forward-class-list        Unconfigured    N/A       None
BGPLU                         Unconfigured    N/A       None
```

```
LPTS ACL                      Unconfigured    N/A       None
Dark Bandwidth                Unconfigured    N/A       None
SR-OPT Enable                 Configured      No        Reload
IP Redirect Punt              Unconfigured    N/A       None
IPv6 Hop-limit Punt           Unconfigured    N/A       None
MPLS Per Path Stats           Unconfigured    N/A       None
Tunnel TTL Decrement          Unconfigured    N/A       None
High-Scale No-LDP-Over-TE     Unconfigured    N/A       None
Label over TE counters        Unconfigured    N/A       None
Highscale LDPoTE No SRoTE     Unconfigured    N/A       None
LPTS Pifib Entry Counters     Unconfigured    N/A       None

Router# reload location all
Thu Jan 26 20:15:32.557 UTC
Proceed with reload? [confirm] y


Router# show hw-module profile cef

-----------------------------------------------------------------
Knob                          Status          Applied   Action
-----------------------------------------------------------------
CBF Enable                    Unconfigured    N/A       None
CBF forward-class-list        Unconfigured    N/A       None
BGPLU                         Unconfigured    N/A       None
LPTS ACL                      Unconfigured    N/A       None
Dark Bandwidth                Unconfigured    N/A       None
SR-OPT Enable                 Configured      Yes       None
IP Redirect Punt              Unconfigured    N/A       None
IPv6 Hop-limit Punt           Unconfigured    N/A       None
MPLS Per Path Stats           Unconfigured    N/A       None
Tunnel TTL Decrement          Unconfigured    N/A       None
High-Scale No-LDP-Over-TE     Unconfigured    N/A       None
Label over TE counters        Unconfigured    N/A       None
Highscale LDPoTE No SRoTE     Unconfigured    N/A       None
LPTS Pifib Entry Counters     Unconfigured    N/A       None
```

Verification

```
Router# show mpls forwarding labels 19001 detail
Tue Feb  5 19:50:13.992 UTC
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
19001  Pop         SR Pfx (idx 3001)  Hu0/1/0/1    18.0.0.2        0
     Updated: Feb  1 23:07:39.595
     Version: 27, Priority: 1
     Label Stack (Top -> Bottom): { Imp-Null }
     NHID: 0x0, Encap-ID: 0x1380900000002, Path idx: 0, Backup path idx: 0, Weight: 0
     MAC/Encaps: 14/14, MTU: 1500
     Outgoing Interface: HundredGigE0/1/0/1 (ifhandle 0x008000c0)
     Packets Switched: 0

  Traffic-Matrix Packets/Bytes Switched: 0/0
  Total Packets/Bytes Switched: 6592788/843876864
```

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
  !
 !
!
```

Circuit-Style SR-TE Policies

Table 11: Feature History Table

Feature Name	Release Information	Feature Description
Circuit-Style SR-TE Policies	Release 7.8.1	This solution allows Segment Routing to meet the requirements of a connection-oriented transport network, which was historically delivered over circuit-switched SONET/SDH networks. Circuit-style SR-TE policies allow a common network infrastructure to be used for both connection-oriented services and classic IP-based transport. This eliminates the need for multiple parallel networks, which greatly reduces both capital expenditures (CapEx) and operating expenditures (OpEx).

Segment Routing provides an architecture that caters to both connectionless transport (such as IP) as well as connection-oriented transport (such as TDM). IP-centric transport uses the benefits of ECMP and automated/optimum protection from TI-LFA. On the other hand, connection-oriented transport, which was historically delivered over circuit-switched SONET/SDH networks, requires the following:

- End-to-end bidirectional transport that provides congruent forward and reverse paths, predictable latency, and disjointness
- Bandwidth commitment to ensure there is no impact on the SLA due to network load from other services
- Monitoring and maintenance of path integrity with end-to-end 50-msec path protection
- Persistent end-to-end paths regardless of control-plane state

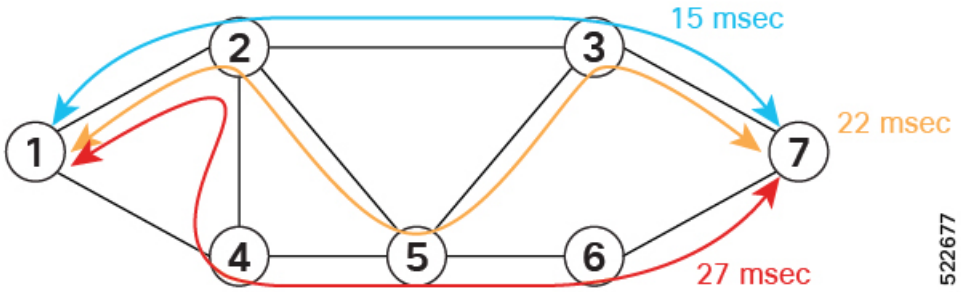
An SR network can satisfy these requirements by leveraging Circuit-Style SR-TE policies (CS-SR policies).

Properties of Circuit-Style SR Policies

CS-SR polices have the following properties:

- **Guaranteed Latency over Non-ECMP Paths**

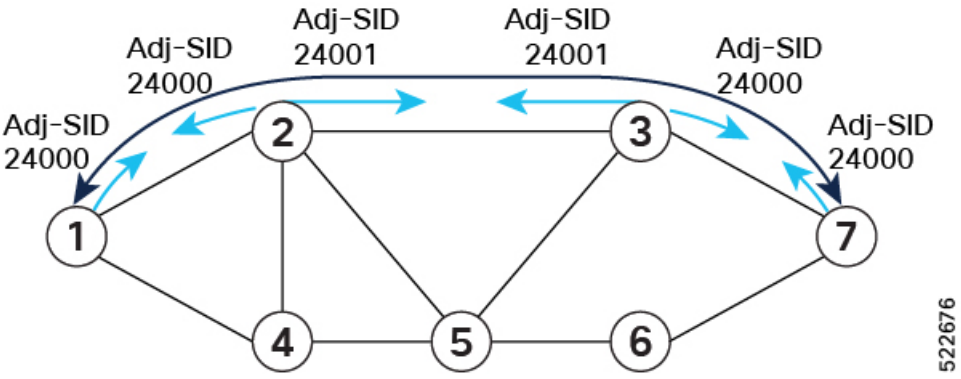
  Consider the network below with three possible paths from node 1 to node 7. Of the three paths, the best end-to-end delay is provided by the blue path (1 -> 2 -> 3 -> 7). The chosen path is then encoded with Adj-SIDs corresponding to the traversed interfaces to avoid any ECMP, and therefore guarantee the latency over the path.



- **Control-Plane Independent Persistency**

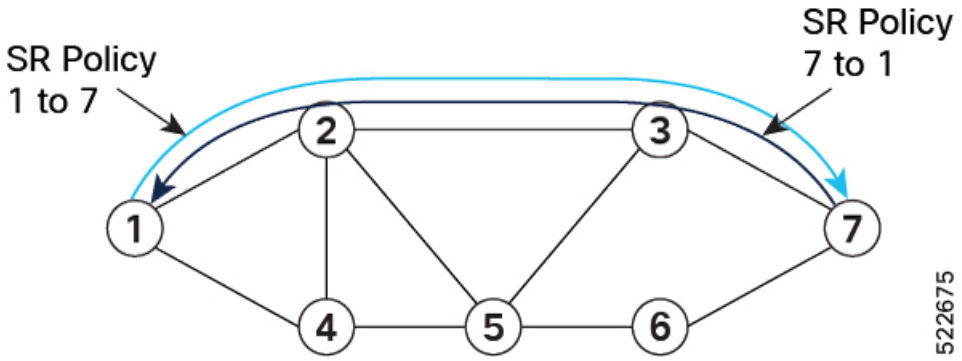
  Adjacency SIDs can provide a persistent path independent from control-plane changes (such as IGP neighbor flaps), as well as network events (such as interface additions or interface flaps) and even the presence of IP on an interface. To achieve this, adjacency SIDs can be manually allocated to ensure persistent values, for example after a node reload event. In addition, adjacency SIDs can be programmed as non-protected to avoid any local TI-LFA protection.

  With the Adj-SIDs depicted in the figure below, the path from node 1 to node 7 is encoded with the segment list of {24000, 24001, 24000}. By manually allocating the same Adj-SID values for other direction, the path from node 7 to node 1 is encoded with the same segment list of {24000, 24001, 24000}.



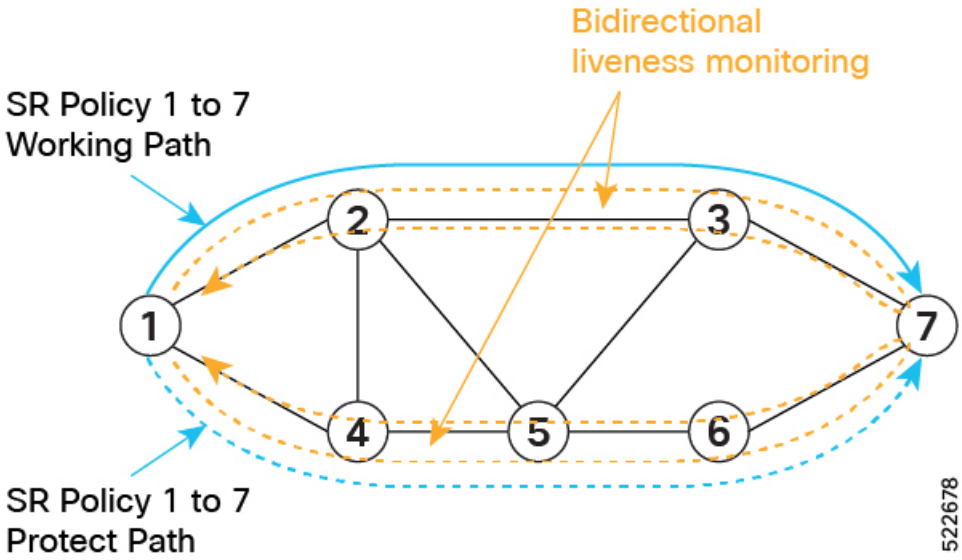
- **Co-Routed Bidirectional Path**

  Forward and return SR Policies with congruent paths are routed along the same nodes/interfaces.



- **Liveness Monitoring with Path Protection Switching**

  Bi-directional liveness monitoring on the working and protect paths ensures fast and consistent switchover, while a protect path is pre-programmed over disjoint facilities.



- **Guaranteed Bandwidth**

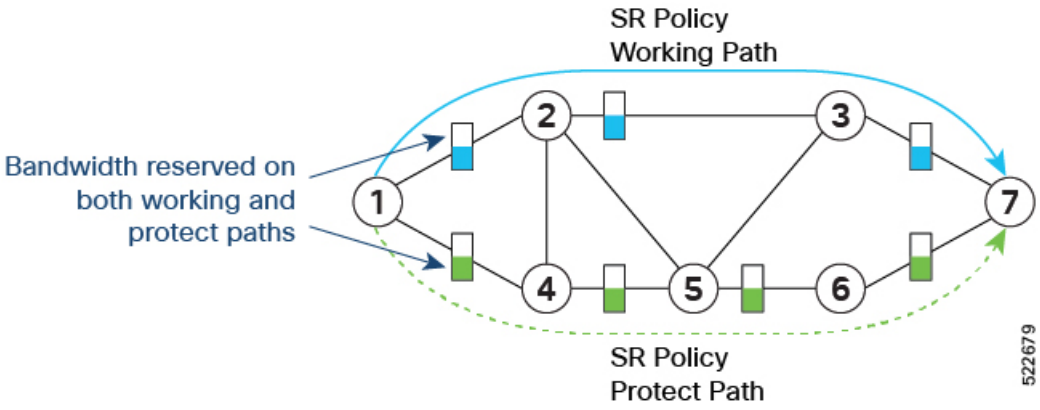
  Most services carried over the CS-SR policy are constant-rate traffic streams. Any packet loss due to temporary congestion leads to bit errors at the service layer. Therefore, bandwidth must be managed very tightly and guaranteed to the services mapped to CS-SR policies.

  A centralized controller manages the bandwidth reservation. The controller maintains the reserved bandwidth on each link based on the traffic usage:

  - Monitors amount of traffic forwarded to each CS-SR policy in the network
  - Uses knowledge of the active path used by the policy
  - Computes the per-link reservable bandwidth accordingly

A per-hop behavior (as documented in RFC3246 [Expedited Forwarding] or RFC2597 [Assured Forwarding]) ensures that the specified bandwidth is available to CS-SR policies at all times independent of any other traffic.

Bandwidth is reserved on both the working and protect paths.



In addition, you can allocate one MPLS-EXP value for traffic steered over the CS SR-TE polices and use QoS (interface queueing) configuration to isolate the circuit traffic from the rest:

- QoS on headend nodes:
  - Define EXP value associated with CS services
  - Enforce rate limiting and perform EXP marking on service ingress interfaces
- QoS on transit nodes:
  - Classify incoming packets based on EXP value associated with CS services.
  - Enforce guaranteed bandwidth for the classified traffic on egress interfaces using bandwidth queues or priority queue with shaper.

Refer to Classify Packets to Identify Specific Traffic chapter in the *Modular QoS Configuration Guide for Cisco 8000 Series Routers*.

Components of the Circuit-Style Solution

CS-SR policy paths are computed and maintained by a stateful PCE. The stateful PCE has a centralized view of the network that it can leverage to compute the co-routed bidirectional end-to-end paths and perform bandwidth allocation control, as well as monitor capabilities to ensure SLA compliance for the life of the CS-SR Policy.

- Centralized Controller
  - Computes the path
  - Encodes the path in a list of Adj-SIDs
  - Monitors and controls bandwidth for SLA guarantee

- QoS configuration on every link to isolate guaranteed traffic

Usage Guidelines and Limitations

Observe the following guidelines and limitations:

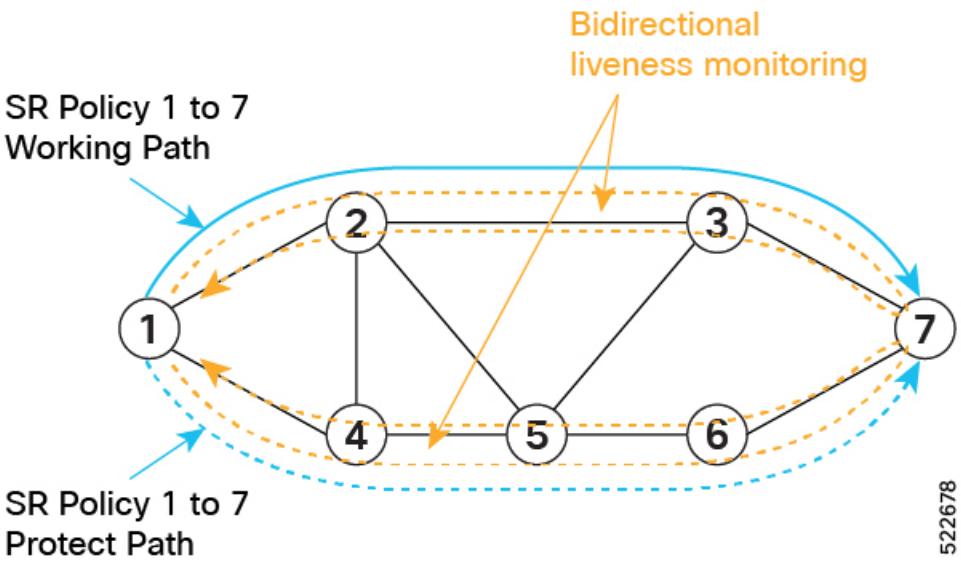
- The maximum SID depth (MSD) is 8.
- CS SR policy end-point IP address must be the router-ID of the intended node.
- SR policy path protection is required for both directions.
- SR policy with dynamic path bandwidth constraint is required for both directions and must have the same value for both directions.
- Candidate path (CP) behavior:
  - The working path is associated with the candidate path of the highest preference value.
  - The protect path is associated with the candidate path of the second-highest preference value.
  - The restore path is associated with the candidate path of the third-highest preference value and is configured as "backup ineligible".
  - Candidate paths with the same role in both directions (working, protect, restore) must have the same preference value.
- Bi-directional path behavior:
  - All paths must be configured as co-routed.
  - All paths with the same role in both directions (working, protect, restore) must have the same bi-directional association ID value.
  - The bi-directional association ID value must be globally unique.
- Disjointness constraint:
  - The working and protect paths under the CS SR policy must be configured with a disjointness constraint using the same disjoint association ID and disjointness type.
    - The disjointness association ID for a working and protect path pair in one direction must be globally unique from the corresponding working and protect path pair in the opposite direction.
  - Node and link disjoint constraint types are supported.
  - The disjoint type used in both directions must be the same.
  - The restore path must not be configured with a disjointness constraint.
- Path optimization objectives supported are TE, IGP, and latency.
- The path optimization objective must match across working, protect, and restore paths in both directions.
- Segment type constraint:
  - Working, protect, and restore paths must all be configured with unprotected-only segment type constraint.

- Working, protect, and restore paths must all be configured with Adj-SID-only segment type constraint.
- To ensure persistency throughout link failure events, manual adjacency SIDs allocated from the SRLB range should be created on all interfaces used by CS policies.

- Revert/recovery behavior:
  - When both working and protect paths are down, the restore path becomes active.
  - The restore path remains active until the working or protect path recovers (partial recovery) and the lock duration timer expires.
  - The lock duration timer is configured under the protect and restore CPs.
- The following functionalities are not supported:
  - Affinity constraint
  - Flex-Algo constraint
  - Metric-bounds constraint

Configure Performance Measurement

Performance Measurement (PM) provides proper detection of candidate path liveness and effective path protection. See SR Policy Liveness Monitoring.



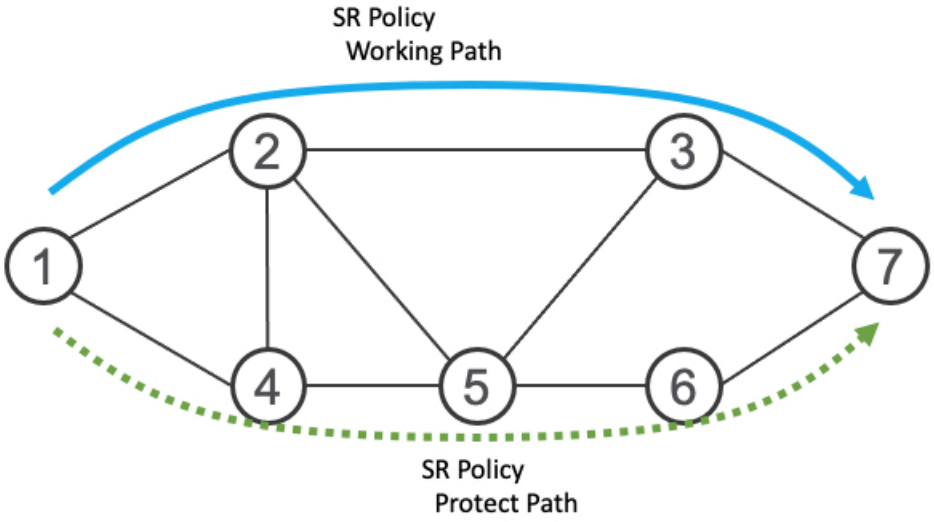
The following example shows how to create a liveness profile for the working and protect paths.

```
Router_1(config)# performance-measurement
Router_1(config-perf-meas)# liveness-profile name profile-WORKING
Router_1(config-pm-ld-profile)# liveness-detection
Router_1(config-pm-ld-profile-ld)# multiplier 3
Router_1(config-pm-ld-profile-ld)# exit
Router_1(config-pm-ld-profile)# probe
```

```
Router_1(config-pm-ld-probe)# tx-interval 30000
Router_1(config-pm-ld-probe)# exit
Router_1(config-pm-ld-profile)# exit
Router_1(config-perf-meas)# liveness-profile name profile-PROTECT
Router_1(config-pm-ld-profile)# liveness-detection
Router_1(config-pm-ld-profile-ld)# multiplier 3
Router_1(config-pm-ld-profile-ld)# exit
Router_1(config-pm-ld-profile)# probe
Router_1(config-pm-ld-probe)# tx-interval 100000
```

Configuring CS SR-TE Policy

The following example shows how to configure a circuit-style SR policy from node 1 to node 7 with three candidate paths: working, protect, and restore.



Create the SR-TE Policy

Configure the CS SR-TE policy

Use the **bandwidth** *bandwidth* command in SR-TE policy configuration mode to configure the guaranteed reservable bandwidth for the policy. The range for *bandwidth* is from 1 to 4294967295 in kbps.

Use the **path-protection** command in SR-TE policy configuration mode to enable end-to-end path protection.

```
Router_1(config)# segment-routing
Router_1(config-sr)# traffic-eng
Router_1(config-sr-te)# policy cs-srte-to-node7
Router_1(config-sr-te-policy)# bandwidth 10000
Router_1(config-sr-te-policy)# color 10 end-point ipv4 10.1.1.7
Router_1(config-sr-te-policy)# path-protection
Router_1(config-sr-te-path-pref-protection)# exit
Router_1(config-sr-te-policy)#
```

Enable Liveness Monitoring Under SR Policy

The following example shows how to enable liveness monitoring under SR Policy, associate the working and protect (backup) liveness-profiles, and configure the invalidation action.

```
Router_1(config)# segment-routing
Router_1(config-sr)# traffic-eng
Router_1(config-sr-te)# policy cs-srte-to-node7
Router_1(config-sr-te-policy)# performance-measurement
Router_1(config-sr-te-policy-perf-meas)# liveness-detection
Router_1(config-sr-te-policy-live-detect)# liveness-profile name profile-WORKING
Router_1(config-sr-te-policy-live-detect)# liveness-profile backup name profile-PROTECT
Router_1(config-sr-te-policy-live-detect)# exit
Router_1(config-sr-te-policy-perf-meas)# exit
Router_1(config-sr-te-policy)#
```

Configure the Working Candidate Path

The working CP has the following characteristics:

- The working CP has the highest preference.
- The working CP uses unprotected-only Adj-SIDs in the segment list.
- The working CP is bidirectional and co-routed.
- The working CP in both directions must have the same bi-directional association ID value.
- The disjoint path constraint for the working CP must have the same group ID and disjoint type as the protect CP.

```
Router_1(config)# segment-routing
Router_1(config-sr)# traffic-eng
Router_1(config-sr-te)# policy cs-srte-to-node7
Router_1(config-sr-te-policy)# candidate-paths
Router_1(config-sr-te-policy-path)# preference 100
Router_1(config-sr-te-policy-path-pref)# dynamic
Router_1(config-sr-te-pp-info)# pcep
Router_1(config-sr-te-path-pcep)# exit
Router_1(config-sr-te-pp-info)# metric
Router_1(config-sr-te-path-metric)# type te
Router_1(config-sr-te-path-metric)# exit
Router_1(config-sr-te-pp-info)# exit
Router_1(config-sr-te-policy-path-pref)# constraints
Router_1(config-sr-te-path-pref-const)# segments
Router_1(config-sr-te-path-pref-const-seg)# protection unprotected-only
Router_1(config-sr-te-path-pref-const-seg)# adjacency-sid-only
Router_1(config-sr-te-path-pref-const-seg)# exit
Router_1(config-sr-te-path-pref-const)# disjoint-path group-id 3 type node
Router_1(config-sr-te-path-pref-const)# exit
Router_1(config-sr-te-policy-path-pref)# bidirectional
Router_1(config-sr-te-path-pref-bidir)# co-routed
Router_1(config-sr-te-path-pref-bidir)# association-id 1100
Router_1(config-sr-te-path-pref-bidir)# exit
Router_1(config-sr-te-policy-path-pref)# exit
Router_1(config-sr-te-policy-path)#
```

Configure the Protect Candidate Path

The protect CP has the following characteristics:

- The protect path is associated with the candidate path of the second-highest preference.
- The protect CP uses unprotected-only Adj-SIDs in the segment list.

- The protect CP is bidirectional and co-routed.
- The protect CP in both directions must have the same bi-directional association ID value.
- The disjoint path constraint for the protect CP must have the same group ID and disjoint type as the working CP.
- When the working path is invalid, the protect path becomes active. After the working path has recovered, the protect path remains active until the default lock duration (300 seconds) expires. You can configure a different lock duration using the **lock duration** *duration* command. The *duration* range is 0 (disabled) to 3000 seconds. If the lock duration is 0 (disabled), then the working path becomes active as soon as it recovers. If *duration* is not specified, the protect path remains active.

```
Router_1(config-sr-te-policy-path)# preference 50
Router_1(config-sr-te-policy-path-pref)# dynamic
Router_1(config-sr-te-pp-info)# pcep
Router_1(config-sr-te-path-pcep)# exit
Router_1(config-sr-te-pp-info)# metric
Router_1(config-sr-te-path-metric)# type te
Router_1(config-sr-te-path-metric)# exit
Router_1(config-sr-te-pp-info)# exit
Router_1(config-sr-te-policy-path-pref)# lock duration 30
Router_1(config-sr-te-policy-path-pref)# constraints
Router_1(config-sr-te-path-pref-const)# segments
Router_1(config-sr-te-path-pref-const-seg)# protection unprotected-only
Router_1(config-sr-te-path-pref-const-seg)# adjacency-sid-only
Router_1(config-sr-te-path-pref-const-seg)# exit
Router_1(config-sr-te-path-pref-const)# disjoint-path group-id 3 type node
Router_1(config-sr-te-path-pref-const)# exit
Router_1(config-sr-te-policy-path-pref)# bidirectional
Router_1(config-sr-te-path-pref-bidir)# co-routed
Router_1(config-sr-te-path-pref-bidir)# association-id 1050
Router_1(config-sr-te-path-pref-bidir)# exit
Router_1(config-sr-te-policy-path-pref)# exit
Router_1(config-sr-te-policy-path)#
```

Configure the Restore Candidate Path

The restore CP has the following characteristics:

- The restore path is associated with the candidate path of the the third-highest preference.
- The restore CP uses unprotected-only Adj-SIDs in the segment list.
- The restore CP is bidirectional and co-routed.
- The restore CP in both directions must have the same bidirectional association ID value.
- The restore CP must be configured with **backup-ineligible**. This configuration prevents the restore CP from being used as a fast reroute backup. The restore path is not computed until both working and protect paths become unavailable.
- Disjointness constraint is not configured on the restore CP.
- If both working and protect paths are unavailable, the restore path becomes active. After either the working or protect path has recovered, the restore path remains active until the default lock duration (300 seconds) expires. You can configure a different lock duration using the **lock duration** *duration* command. The *duration* range is 0 (disabled) to 3000 seconds. If the lock duration is 0 (disabled), then the working

or protect path becomes active as soon as either recovers. If *duration* is not specified, the restore path remains active.

```
Router_1(config-sr-te-policy-path)# preference 10
Router_1(config-sr-te-policy-path-pref)# dynamic
Router_1(config-sr-te-pp-info)# pcep
Router_1(config-sr-te-path-pcep)# exit
Router_1(config-sr-te-pp-info)# metric
Router_1(config-sr-te-path-metric)# type te
Router_1(config-sr-te-path-metric)# exit
Router_1(config-sr-te-pp-info)# exit
Router_1(config-sr-te-policy-path-pref)# backup-ineligible
Router_1(config-sr-te-policy-path-pref)# lock duration 30
Router_1(config-sr-te-policy-path-pref)# constraints
Router_1(config-sr-te-path-pref-const)# segments
Router_1(config-sr-te-path-pref-const-seg)# protection unprotected-only
Router_1(config-sr-te-path-pref-const-seg)# adjacency-sid-only
Router_1(config-sr-te-path-pref-const-seg)# exit
Router_1(config-sr-te-path-pref-const)# exit
Router_1(config-sr-te-policy-path-pref)# bidirectional
Router_1(config-sr-te-path-pref-bidir)# co-routed
Router_1(config-sr-te-path-pref-bidir)# association-id 1010
Router_1(config-sr-te-path-pref-bidir)# exit
Router_1(config-sr-te-policy-path-pref)# exit
Router_1(config-sr-te-policy-path)#
```

Running Config

```
Router_1# show running-config

. . .
segment-routing
 traffic-eng
  policy cs-srte-to-node7
   bandwidth 10000
   color 10 end-point ipv4 10.1.1.7
   path-protection
   !
   candidate-paths
    preference 10
     dynamic
      pcep
      !
      metric
       type te
      !
     !
     lock
      duration 30
     !
     backup-ineligible
     !
     constraints
      segments
       protection unprotected-only
       adjacency-sid-only
      !
     !
     bidirectional
      co-routed
```

```
      association-id 1010
     !
    !
    preference 50
     dynamic
      pcep
      !
      metric
       type te
      !
     !
     lock
      duration 30
     !
     constraints
      segments
       protection unprotected-only
       adjacency-sid-only
      !
      disjoint-path group-id 3 type node
     !
     bidirectional
      co-routed
      association-id 1050
     !
    !
    preference 100
     dynamic
      pcep
      !
      metric
       type te
      !
     !
     constraints
      segments
       protection unprotected-only
       adjacency-sid-only
      !
      disjoint-path group-id 3 type node
     !
     bidirectional
      co-routed
      association-id 1100
     !
    !
   !
   performance-measurement
    liveness-detection
     liveness-profile backup name profile-PROTECT
     liveness-profile name profile-WORKING
     invalidation-action down
    !
   !
  !
 !
!
root
performance-measurement
 liveness-profile name profile-PROTECT
  liveness-detection
   multiplier 3
  !
  probe
```

```
   tx-interval 100000
  !
 !
 liveness-profile name profile-WORKING
  liveness-detection
   multiplier 3
  !
  probe
   tx-interval 30000
  !
 !
!
```

Verification

Use the **show segment-routing traffic-eng policy detail** command to display the details of the CS SR policy:

```
Router_1# show segment-routing traffic-eng policy detail

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

Color: 10, End-point: 10.1.1.7
  Name: srte_c_10_ep_10.1.1.7
  Status:
    Admin: up  Operational: up for 00:02:24 (since Nov 30 08:03:36.588)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Name: cs-srte-to-node7
      Requested BSID: 8000
      PCC info:
        Symbolic name: cfg_cs-srte-to-node7_discr_100
        PLSP-ID: 2
      Constraints:
        Protection Type: unprotected-only
        Maximum SID Depth: 8
        Adjacency SIDs Only: True
      Performance-measurement:
        Reverse-path Label: Not Configured
        Delay-measurement: Disabled
        Liveness-detection: Enabled
          Profile: profile-WORKING
          Invalidation Action: down
          Logging:
            Session State Change: No
        Statistics:
          Session Create    : 1
          Session Update    : 12
          Session Delete    : 4
          Session Up        : 8
          Session Down      : 3
          Delay Notification: 0
          Session Error     : 0
      Dynamic (pce 192.168.0.5) (valid)
        Metric Type: TE,   Path Accumulated Metric: 10
          SID[0]: 24001 [Adjacency-SID, 10.10.10.1 - 10.10.10.2]
        Reverse path:
          SID[0]: 24000 [Adjacency-SID, 10.10.10.2 - 10.10.10.1]
      Protection Information:
        Role: WORKING
        Path Lock: Timed
        Lock Duration: 300(s)
    Preference: 50 (configuration) (protect)
```

```
    Name: cs-srte-to-node7
    Requested BSID: 8000
    PCC info:
      Symbolic name: cfg_cs-srte-to-node7_discr_50
      PLSP-ID: 1
    Constraints:
      Protection Type: unprotected-only
      Maximum SID Depth: 8
      Adjacency SIDs Only: True
    Performance-measurement:
      Reverse-path Label: Not Configured
      Delay-measurement: Disabled
      Liveness-detection: Enabled
        Profile: profile-PROTECT
        Invalidation Action: down
        Logging:
          Session State Change: No
      Statistics:
        Session Create    : 0
        Session Update    : 9
        Session Delete    : 0
        Session Up        : 1
        Session Down      : 0
        Delay Notification: 0
        Session Error     : 0
    Dynamic (pce 192.168.0.5) (valid)
      Metric Type: TE,   Path Accumulated Metric: 10
        SID[0]: 24002 [Adjacency-SID, 11.11.11.1 - 11.11.11.2]
      Reverse path:
        SID[0]: 24003 [Adjacency-SID, 11.11.11.2 - 11.11.11.1]
    Protection Information:
      Role: PROTECT
      Path Lock: Timed
      Lock Duration: 30(s)
  Preference: 10 (configuration) (inactive)
    Name: cs-srte-to-node7
    Requested BSID: 8000
    Constraints:
      Protection Type: unprotected-only
      Maximum SID Depth: 8
      Adjacency SIDs Only: True
    Performance-measurement:
      Reverse-path Label: Not Configured
      Delay-measurement: Disabled
      Liveness-detection: Enabled
        Profile: working
        Invalidation Action: down
        Logging:
          Session State Change: No
      Statistics:
        Session Create    : 0
        Session Update    : 0
        Session Delete    : 0
        Session Up        : 0
        Session Down      : 0
        Delay Notification: 0
        Session Error     : 0
    Dynamic (pce) (inactive)
      Metric Type: TE,   Path Accumulated Metric: 0
    Protection Information:
      Role: RESTORE
      Path Lock: Timed
      Lock Duration: 30(s)
LSPs:
```

```
  LSP[0]:
    LSP-ID: 3 policy ID: 1 (standby)
    Local label: 24037
    State: Standby programmed state
    Performance-measurement:
      Reverse-path Label: Not Configured
      Delay-measurement: Disabled
      Liveness-detection: Enabled
        Profile: profile-WORKING
        Invalidation Action: down
        Logging:
          Session State Change: No
        Session State: up, for 1d12h (since Nov 30 08:03:37.859)
  LSP[1]:
    LSP-ID: 7 policy ID: 1 (active)
    Local label: 24036
    State: Programmed
    Binding SID: 8000
    Performance-measurement:
      Reverse-path Label: Not Configured
      Delay-measurement: Disabled
      Liveness-detection: Enabled
        Profile: profile-WORKING
        Invalidation Action: down
        Logging:
          Session State Change: No
        Session State: up, for 05:42:36 (since Dec  1 15:11:36.203)
Attributes:
  Binding SID: 8000
  Forward Class: Not Configured
  Steering labeled-services disabled: no
  Steering BGP disabled: no
  IPv6 caps enable: yes
  Bandwidth Requested: 10000 kbps
  Bandwidth Current: 10000 kbps
  Invalidation drop enabled: no
  Max Install Standby Candidate Paths: 0
```

Reporting of SR-TE Policies Using BGP- Link State

Table 12: Feature History Table

Feature Name	Release Information	Feature Description
Reporting of SR-TE Policies Using BGP-Link State	Release 24.1.1	BGP- Link State (LS) is a mechanism by which LS and Traffic Engineering (TE) information can be collected from networks and shared with external components (such as, Segment Routing Path Computation Element (SR-PCE) or Crossword Optimization Engine (COE)) using the BGP routing protocol. This feature gathers the Traffic Engineering Policy information that is locally available in a node and advertises it in BGP-LS for SR-MPLS and SRv6. The operators can now take informed decisions based on the information that is gathered on their network's path computation, reoptimization, service placement, network visualization, and so on. The feature introduces these changes: **CLI:** • **distribute link-state** **YANG Data Model:** • New XPaths for `module Cisco-IOS-XR-infra-xtc-agent-cfg.yang` (see GitHub, YANG Data Models Navigator)

This function is achieved using a BGP Network Layer Reachability Information (NLRI) encoding format. BGP-LS consumes structured IGP data (for example, router-id, remote-IP-address of a link, local-IP address of a link, link identifier, and so on). and creates BGP-LS (NLRI) or attributes that BGP or other components like Cisco IOS XR Traffic Controller (XTC) can consume. Current implementation of BGP-LS can report topology using Nodes, Links, and Prefixes.

Modern Segment Routing (SR) networks often use SR Traffic Engineering (SR-TE) to influence the path that each specific traffic takes over the network. SR-TE tunnels can be provisioned manually on the tunnel head, but often they are calculated and provisioned by the central controller. Often operator of the network wants the ability to force the traffic over specific nodes and links.

Now the operators have the option to collect reports of the SR-TE and Policy information that is locally available in a node and advertise it into BGP-LS updates, which can be used by external components. Refer the IEFT for examples. This feature is implemented so that the operators have control over their network's path computation, reoptimization, service placement, network visualization, and so on.

Note Circuit Style (CS) SR policies are reported but without the CS policies' specific attributes, like the bidirectional constraints, per-hop behavior, and so on.

Restrictions to Reporting of SRTE Policies using BGP-LS

Some important points to be aware related to this feature are as follows:

- This feature has high availability, ensuring BGP-LS reporting at all times.
- This feature works with PCE State Sync. The policy information learned via the state-sync channel is not advertised in BGP-LS by the PCE.
- The feature works with PCE redundancy. Both PCEs advertise the BGP-LS policy information. The consumers can select only one policy based on the BGP best path logic.

Configure Reporting of SRTE Policies using BGP-LS

The reporting of policies to BGP-LS is disabled by default. Configuring the **distribute link-state** under the SR-PCE or SR-TE configuration enables this feature. Once enabled, all the existing SR policies or Candidate Path (CPs) are encoded to BGP-LS. You can disable this feature by removing the configuration and all the SR policies or CPs are withdrawn from BGP which deletes all of the previously encoded SR policies or CPs.

Note When SR policies that are reporting in BGP-LS are enabled by the operator, the Head End only reports the Active Candidate Path (CPs) (CPs installed in the forwarding). There are monitoring use cases that require reporting of inactive CPs. The following CLI is needed to report inactive CPs.

For both SR-TE and SR-PCE, you need to use the global CLI to enable the reporting of policies or Circuit Style (CS) to BGP-LS:

Configuration Example

Configure the following command to enable reporting and syncing of SR policies in BGP-LS at the Head End:

```
Router# config
Router(config)# segment-routing
Router(config-sr)# traffic-eng
Router(config-sr-te)# distribute link-state
Router(config-sr-te-distribute-ls)# report-candidate-path-inactive
Router(config-sr-te-distribute-ls)# commit
Router(config-sr-te-distribute-ls)# exit
```

Note The **report-candidate-path-inactive** command is an optional command to report inactive CPs.

Running Configuration

This is a sample running configuration which shows that you have configured BGP-LS reporting feature.

```
segment-routing
 traffic-eng
  distribute link-state
   report-candidate-path-inactive
  !
 !
!
```

Verification

Use the **show pce segment-routing traffic-eng policy private** and **show pce distributed-ls events** to verify the configuration.

```
Router#show pce segment-routing traffic-eng policy private

PCE's policy database:
----------------------
PCC Address: 192.168.2.1
  Color: 100, Endpoint: 192.0.2.1
    Name: srte_c_100_ep_192.0.2.1
    Self Pointer: 0x317d7f0
    Active C-path Pointer: 0x317d9b0
    Candidate-paths:
      Symbolic-name: cfg_foo_discr_100 (Active)
        PLSP-ID: 1
        NB-API Notification pending flag: 0
        Self Pointer: 0x317d9b0
        Tunnel Pointer: 0x317d4e0
        Policy Pointer: 0x317d7f0
        Cached distribute LS element:
          Sense: TRUE
          Refcount: 1
          Is on queue: FALSE
          Node identifiers:
            Protocol: 9 router ID: 192.0.2.1
          Policy identifiers:
            Color: 100 Endpoint: 192.0.2.1
            Flags: 0x00
          Candidate path identifiers:
            Originator: 0.0.0.0 protocol: 3 ASN: 0
            Flags: 0x00
            Discriminator: 100
          Candidate path attributes:
            Name: foo
            Policy name: srte_c_100_ep_192.0.2.1
            State:
              Priority: 0 flags: 0x5800 (active, evaluated, valid-sid-list)
              Preference: 100
            BSID:
              Flags: 0x4000 (alloc)
              BSID: 24021
              Specified BSID: 0
            Constraints:
              Bitfield: 0x0020 flags: 0x4000 (protected-only) MT-ID: 0
              Algorithm: 0
              Bandwidth: 0 kbps
              Metric constraints[0]:
                Type: 0 flags: 0x80 (optimization)
                Margin: 0 bound: 0
              Metric constraints[1]:
                Type: 4 flags: 0x10 (bound)
                Margin: 0 bound: 10
```

```
           Segment lists:
             Segment list[0]:
               Flags: 0x3800 (computed, verified, first-seg-resolved) MT-ID: 0 algorithm:
 0 weight: 0
               Metric[0]:
                 Type: 0 flags: 0x10 (value)
                 Margin: 0 bound: 0 value: 10
               Segments:
                 Segment[0]:
                   Bitfield: 0x0000 type: 3 flags: 0x8000 (sid-present)
                   SID: 102000
                   Descriptor:
                     Algorithm: 0
                     Local address: 192.0.2.1 [0] remote address: 0.0.0.0 [0]
```

```
Router#show pce distribute-ls events

Distribute LS events:
----------------
Event history (oldest first):
  Time                    Event
  Apr 11 01:50:48.985     [UID: 0] SR CP NLRI: node ID: SR RID 192.0.2.1, ls_id 0, asn 0,
bitf 0x00000200 policy ID: endpoint: 192.168.0.2 color: 100 CP ID: originator: config 0.0.0.0
 asn: 0 discriminator: 100 (oper: add)
  Apr 11 01:50:50.046     [UID: 1] SR CP NLRI: node ID: SR RID 192.0.2.1, ls_id 0, asn 0,
bitf 0x00000200 policy ID: endpoint: 192.168.0.2 color: 100 CP ID: originator: config 0.0.0.0
 asn: 0 discriminator: 100 (oper: add)
RP/0/0/CPU0:rtrX#show pce distribute-ls summary
Tue Apr 11 02:03:36.289 PDT
Distribution enabled: yes
Connected to LS-LIB: yes
Encode queue size: 0
Estimated encoding rate: 17280/s
NLRIs encoded to LS-LIB:
  SR candidate path: 2 added, 0 removed, 0 errored, 0 replaced
Element stats:
  SR candidate path: 1 (watermark: 2)
Throttle timer:
  Running: no
```

Below is the example show output for SRv6.

```
Router# show segment-routing traffic-eng policy color 200 private

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

Color: 200, End-point: 192::2 ID: 2
  Name: srte_c_200_ep_192::2
  Status:
    Admin: up  Operational: up for 00:01:07 (since Mar  6 11:17:42.580)
  Candidate-paths:
    Preference: 100 (configuration) (active)
      Originator: ASN 0 node-address <None> discriminator: 100
      Name: bar
      Requested BSID: dynamic
      PCC info:
        Symbolic name: cfg_bar_discr_100
        PLSP-ID: 2
        Is orphan: no
        State timer:
          Running: no
      Constraints:
        Protection Type: protected-preferred
```

```
      Maximum SID Depth: 10
    ID: 1
    Source: 192::1
    Stale: no
    Checkpoint flags: 0x00000000
    Path Type: SRV6
    Performance-measurement:
      Reverse-path segment-list:
      Delay-measurement: Disabled
      Liveness-detection: Disabled
    Dynamic (pce 192.168.0.3) (valid)
      Metric Type: IGP,   Path Accumulated Metric: 10
      IGP area: 0
        SID[0]: fccc:ccc1:2::/48 Behavior: uN (PSP/USD) (48)
                Format: f3216
                LBL:32 LNL:16 FL:0 AL:80
                Address: 192::2
    SRv6 Information:
      Locator: loc1Algo0
      Binding SID requested: Dynamic
      Binding SID behavior: uB6 (Insert.Red)
    Cached distribute LS element:
      Sense: TRUE
      Refcount: 1
      Is on queue: FALSE
      Node identifiers:
        Protocol: 9 router ID: 192::1
      Policy identifiers:
        Color: 200 Endpoint: 192::2
        Flags: 0x80 (endpoint-v6)
      Candidate path identifiers:
        Originator: 0.0.0.0 protocol: 3 ASN: 0
        Flags: 0x00
        Discriminator: 100
      Candidate path attributes:
        Name: bar
        Policy name: srte_c_200_ep_192::2
        State:
          Priority: 0 flags: 0x5A00 (active, evaluated, valid-sid-list, delegated)
          Preference: 100
        SRv6 BSID:
          Flags: 0x8000 (alloc)
          BSID: fccc:ccc1:1:e018::
          Specified BSID: ::
          Endpoint:
            Endpoint function: 71 flags: 0x00 algorithm: 0
          Structure:
           Locator block length: 32 locator node length: 16 function length: 16 arguments
 length: 0
        Constraints:
          Bitfield: 0x0020 flags: 0xC000 (dataplane-v6, protected) MT-ID: 0
          Algorithm: 0
          Bandwidth: 0 kbps
          Metric constraints[0]:
            Type: 0 flags: 0x80 (optimization)
            Margin: 0 bound: 0
          Metric constraints[1]:
            Type: 4 flags: 0x10 (bound)
            Margin: 0 bound: 10
        Segment lists:
          Segment list[0]:
            Flags: 0xB800 (dataplane-v6, computed, verified, first-seg-resolved) MT-ID:
 0 algorithm: 0 weight: 1
```

```
                Metric[0]:
                  Type: 0 flags: 0x10 (value)
                  Margin: 0 bound: 0 value: 10
                Segments:
                  Segment[0]:
                    Bitfield: 0x0003 type: 9 flags: 0x8000 (sid-present)
                    SID: fccc:ccc1:2::
                    Descriptor:
                      Algorithm: 0
                      Local address: 192::2 [0] remote address: :: [0]
                    Endpoint:
                      Endpoint function: 48 flags: 0x00 algorithm: 0
                    Structure:
                      Locator block length: 32 locator node length: 16 function length: 0
arguments length: 80
  LSPs:
    LSP[0]:
      LSP-ID: 2 policy ID: 2 (active)
      State: Programmed
      Binding SID: fccc:ccc1:1:e018::
      Install timer:
        Running: no
      Cleanup timer:
        Running: no
      Delete timer:
        Running: no
      Revert timer:
        Running: no
      SM chain:
        Init -> Egress paths
        Egress paths pending -> BSID RW
        BSID rewrite pending -> Success
      Forwarding flags: 0x00000008
      Candidate path ID: 1
      Flags:
      SL-ID:
        Sent/Received/Transferred: 1/1/0
      SLs:
        SL[0]:
          Name: dynamic
          Type: Dynamic PCE
          Checkpoint id: 1
          NH SRV6 SID: fccc:ccc1:2::
          SL ID: 0xa000001
          Flags:
          Normalized Weight: 1
          ENS: 1
          Paths:
            Path[0]:
              Interface version: 1
              Flags:
              Outgoing interface: Gi0/2/0/0
              Weight: 1
              SID stack:
              Total SID count: 0
              Underlay Normalized Weight: 1
            Path[1]:
              Interface version: 1
              Flags:
              Outgoing interface: Gi0/2/0/1
              Weight: 1
              SID stack:
              Total SID count: 0
              Underlay Normalized Weight: 1
```

```
            Path[2]:
              Interface version: 1
              Flags:
              Outgoing interface: Gi0/2/0/2
              Weight: 1
              SID stack:
              Total SID count: 0
              Underlay Normalized Weight: 1
  Attributes:
    Binding SID: fccc:ccc1:1:e018::
    Forward Class: Not Configured
    Steering labeled-services disabled: no
    Steering BGP disabled: no
    IPv6 caps enable: yes
    Invalidation drop enabled: no
    Max Install Standby Candidate Paths: 0
    Path Type: SRV6
  Notification to clients:
    Binding SID: fccc:ccc1:1:e018::
    Bandwidth : 0 Kbps (0 Kbps)
    State: UP
    Flags: [add] [ipv6_caps]
    Metric Type: IGP
    Metric Value: 10
    Admin Distance: 30
  ifhandle: 0x00000000
  Source: 192::1
  Transition count: 1
  LSPs created count: 1
  Reoptimizations completed count: 1
  Retry Action Flags: 0x00000000, ()
  Last Retry Timestamp: never (0 seconds ago)
  Policy reference: 0x1222c10
  Event history (oldest first):
    Time                    Event
    Mar  6 11:17:40.563     POLICY CREATE
    Mar  6 11:17:40.564     (x3) CP PCRPT: 1 hops:
    Mar  6 11:17:41.071     CP PCUPD: CP-ID: 1, path change: true, notify: true, hops:
fccc:ccc1:2::
    Mar  6 11:17:41.072     CP PCRPT: 1 hops: fccc:ccc1:2::
    Mar  6 11:17:41.072     LSP CREATE: 2, need BSID RW: true, BSID: ::
    Mar  6 11:17:41.072     CP CHANGE: PREF: 100 [PROTO: 30, ORIGIN: 0/<None>, DISC: 100]
    Mar  6 11:17:42.579     SRv6 BSID RW REQ: ID 2
    Mar  6 11:17:42.580     SRv6 BSID RW RES: ID 2 Status: success
    Mar  6 11:17:42.580     LSP PCRPT: 2, hops: fccc:ccc1:2::
    Mar  6 11:17:42.580     CP PCRPT REMOVE: 1
    Mar  6 11:17:42.580     IM STATE CHANGE: UNKNOWN to UP, count: 0


Router# show pce segment-routing traffic-eng policy private

PCE's policy database:
----------------------

PCC Address: 192.168.2.1
  Color: 200, Endpoint: 192::2
    Name: srte_c_200_ep_192::2
    Self Pointer: 0x26cd890
    Active C-path Pointer: 0x26cda00
    Candidate-paths:
      Symbolic-name: cfg_bar_discr_100 (Active)
        PLSP-ID: 2
        NB-API Notification pending flag: 0
```

```
        Self Pointer: 0x26cda00
        Tunnel Pointer: 0x26cd580
        Policy Pointer: 0x26cd890
        Cached distribute LS element:
          Sense: TRUE
          Refcount: 1
          Is on queue: FALSE
          Node identifiers:
            Protocol: 9 router ID: 192::1
          Policy identifiers:
            Color: 200 Endpoint: 192::2
            Flags: 0x80 (endpoint-v6)
          Candidate path identifiers:
            Originator: 0.0.0.0 protocol: 3 ASN: 0
            Flags: 0x00
            Discriminator: 100
          Candidate path attributes:
            Name: bar
            Policy name: srte_c_200_ep_192::2
            State:
              Priority: 0 flags: 0x5A00 (active, evaluated, valid-sid-list, delegated)
              Preference: 100
            SRv6 BSID:
              Flags: 0x8000 (alloc)
              BSID: fccc:ccc1:1:e018::
              Specified BSID: ::
              Endpoint:
                Endpoint function: 71 flags: 0x00 algorithm: 0
              Structure:
               Locator block length: 32 locator node length: 16 function length: 16 arguments
length: 0
            Constraints:
              Bitfield: 0x0020 flags: 0xC000 (dataplane-v6, protected) MT-ID: 0
              Algorithm: 0
              Bandwidth: 0 kbps
              Metric constraints[0]:
                Type: 0 flags: 0x80 (optimization)
                Margin: 0 bound: 0
              Metric constraints[1]:
                Type: 4 flags: 0x10 (bound)
                Margin: 0 bound: 10
            Segment lists:
              Segment list[0]:
               Flags: 0xB800 (dataplane-v6, computed, verified, first-seg-resolved) MT-ID:
0 algorithm: 0 weight: 0
                Metric[0]:
                  Type: 0 flags: 0x10 (value)
                  Margin: 0 bound: 0 value: 10
                Segments:
                  Segment[0]:
                    Bitfield: 0x0003 type: 9 flags: 0x8000 (sid-present)
                    SID: fccc:ccc1:2::
                    Descriptor:
                      Algorithm: 0
                      Local address: 192::2 [0] remote address: :: [0]
                    Endpoint:
                      Endpoint function: 48 flags: 0x00 algorithm: 0
                    Structure:
                      Locator block length: 32 locator node length: 16 function length: 0
arguments length: 80
```

```
Router# show bgp link-state link-state | i \\[SP\\]

*>i[SP][SR][I0x0][N[c100][b0.0.0.0][q192.168.0.1][te192::1]][C[po0x3][f0x80][e192::2][cl0xc8][as0][oa0.0.0.0][di100]]/792
*>
[SP][SR][I0x0][N[c100][b0.0.0.0][q192.168.0.3][te192::1]][C[po0x3][f0x80][e192::2][cl0xc8][as0][oa0.0.0.0][di100]]/792
```

```
Router# show bgp link-state link-state
[SP][SR][I0x0][N[c100][b0.0.0.0][q192.168.0.1][te192::1]][C[po0x3][f0x80][e192::2][cl0xc8][as0][oa0.0.0.0][di100]]/792
BGP routing table entry for
[SP][SR][I0x0][N[c100][b0.0.0.0][q192.168.0.1][te192::1]][C[po0x3][f0x80][e192::2][cl0xc8][as0][oa0.0.0.0][di100]]/792
Versions:
  Process           bRIB/RIB   SendTblVer
  Speaker                128          128
Last Modified: Mar  6 11:17:43.000 for 00:04:09
Last Delayed at: ---
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    192.168.2.1 (metric 20) from 192.168.2.1 (192.168.0.1)
      Origin IGP, localpref 100, valid, internal, best, group-best
      Received Path ID 0, Local Path ID 1, version 128
      Link-state:
                  SRTE-CP-State: Priority: 0 Flags: 0x5a00 Preference: 100
                  SR-Policy-CP-name: bar
                  SR-Policy-name: srte_c_200_ep_192::2
                  SRTE-CP-Constraints: Flags: 0xc000 Mtid: 0 Algorithm: 0
                  SRTE-CP-Constraints-Metric: Type: 0 Flags: 0x80 Margin: 0
                  Bound: 0
                  SRTE-CP-Constraints-Metric: Type: 4 Flags: 0x10 Margin: 0
                  Bound: 10
                  SRTE-Segment-List: Flags: 0xb800 Mtid: 0 Algorithm: 0
                  Weight: 1
                 SRTE-Segment: Segment-Type: 9 Flags: 0x8000 SID: fccc:ccc1:2:: Algorithm:
 0
                   Local-node: 192::2
                  SRv6-Endpoint-Fn: 48 Flags: 0x0 Algo: 0
                  SRv6-SID-Struct: LBL: 32 LNL: 16 FL: 0 AL: 80
                  SRTE-Segment-List-Metric: Type: 0 Flags: 0x10 Margin: 0
                  Bound: 0 Value: 10
                  SRTE-CP-SRV6-BSID: Flags: 0x8000 BSID: fccc:ccc1:1:e018::
                   Specified BSID: ::
                  SRv6-Endpoint-Fn: 71 Flags: 0x0 Algo: 0
                  SRv6-SID-Struct: LBL: 32 LNL: 16 FL: 16 AL: 0
```

SR Policy Path Computation for IPv6

Table 13: Feature History Table

Feature Name	Release Information	Feature Description
SR Policy Path Computation for IPv6	Release 24.1.1	We have introduced the capability for SR Policy to support segment lists with IPv6 addresses, which can be either dynamically computed or explicitly set at the SRTE headend.

You can now use this feature when you want SR Policy to support segment lists with IPv6 addressed.

Note It uses the exiting configurations outlined in *Chapter: Configure SR-TE Policies* in *Segment Routing Configuration Guide for Cisco 8000 Series Routers*, with the supported features detailed in the Usage Guidelines section that follows.

Usage Guidelines and Limitations

The supported features are listed below in the same order as the Table of Contents within the *Chapter: Configure SR-TE Policies* in *Segment Routing Configuration Guide for Cisco 8000 Series Routers*

Supported Features

- Instantiation of SR Policy
  - On-Demand SR Policy – SR On-Demand Next-Hop
  - Manually Provisioned SR Policy
- SR-TE BGP Soft Next-Hop Validation For ODN Policies
- SR-TE Policy Path Types
  - Dynamic Paths
    - Optimization Objectives
    - Constraints
    - Configure SR Policy with Dynamic Path
  - Explicit Paths
    - SR-TE Policy with Explicit Path
    - Explicit Path with Affinity Constraint Validation
    - Explicit Path with Segment Protection-Type Constraint
- Traffic Steering
  - Automated Steering
  - Color-Only Automated Steering
  - Static Route over Segment Routing Policy
- Services Supported
  - IPv4 BGP Global Routes
  - IPv6 BGP Global Routes
- Miscellaneous

- SR-TE Reoptimization Timers
- Sharing the Extended Label Switch Path Array