



# Configure Segment Routing Path Computation Element

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



**Note** The Cisco IOS XRv 9000 is the recommended platform to act as the SR-PCE. Refer to the [Cisco IOS XRv 9000 Router Installation and Configuration Guide](#) for more information.

**Table 1: Feature History Table**

Feature Name	Release Information	Feature Description
SR-PCE: Single PCE scale enhancement	Release 7.5.1	With this feature, support for a single PCE is enhanced to 50000 nodes, 100000 LSPs, 500000 links, and 2000 PCEP sessions.

- [About SR-PCE, on page 2](#)
- [Usage Guidelines and Limitations, on page 3](#)
- [Configure SR-PCE, on page 3](#)
- [PCE override of SR-TE policies, on page 8](#)
- [PCE-Initiated SR Policies, on page 13](#)
- [SR-PCE Flexible Algorithm Multi-Domain Path Computation, on page 15](#)
- [ACL Support for PCEP Connection, on page 19](#)
- [Anycast SID-Aware Path Computation, on page 20](#)
- [SR-PCE IPv4 Unnumbered Interface Support, on page 25](#)
- [Inter-Domain Path Computation Using Redistributed SID, on page 27](#)
- [Configuring the North-Bound API on SR-PCE, on page 30](#)

# About SR-PCE

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
TCP Authentication Option	Release 7.3.1	This feature introduces support for TCP Authentication Option (TCP-AO), which replaces the TCP Message Digest 5 (MD5) option, which was used for authenticating PCEP (TCP) sessions by using a clear text or encrypted password.

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



**Note** For more information on PCE, PCC, and PCEP, refer to the [Path Computation Element](#) section in the *MPLS Configuration Guide for Cisco ASR 9000 Series Routers*.

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

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

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

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

### TCP Authentication Option

TCP Message Digest 5 (MD5) authentication has been used for authenticating PCEP (TCP) sessions by using a clear text or encrypted password. This feature introduces support for TCP Authentication Option (TCP-AO), which replaces the TCP MD5 option.

TCP-AO uses Message Authentication Codes (MACs), which provides the following:

- Protection against replays for long-lived TCP connections
- More details on the security association with TCP connections than TCP MD5
- A larger set of MACs with minimal system and operational changes

TCP-AO is compatible with Master Key Tuple (MKT) configuration. TCP-AO also protects connections when using the same MKT across repeated instances of a connection. TCP-AO protects the connections by using traffic key that are derived from the MKT, and then coordinates changes between the endpoints.



---

**Note** TCP-AO and TCP MD5 are never permitted to be used simultaneously. TCP-AO supports IPv6, and is fully compatible with the proposed requirements for the replacement of TCP MD5.

---

## Usage Guidelines and Limitations

To ensure PCEP compatibility, we recommend that the Cisco IOS XR version on the SR-PCE be the same or later than the Cisco IOS XR version on the PCC or head-end.

## Configure SR-PCE

This task explains how to configure SR-PCE.

### Before you begin

The Cisco IOS XRv 9000 is the recommended platform to act as the SR-PCE.

### SUMMARY STEPS

1. **configure**
2. **pce**
3. **address ipv4** *address*
4. **state-sync ipv4** *address*
5. **tcp-buffer size** *size*
6. **password** {**clear** | **encrypted**} *password*
7. **tcp-ao** *key-chain* [**include-tcp-options**] [**accept-ao-mismatch-connection**]
8. **segment-routing** {**strict-sid-only** | **te-latency**}
9. **timers**
10. **keepalive** *time*
11. **minimum-peer-keepalive** *time*

12. `reoptimization time`
13. `exit`

## DETAILED STEPS

### Procedure

	Command or Action	Purpose
<b>Step 1</b>	<b>configure</b> <b>Example:</b> <code>RP/0/RSP0/CPU0:router# configure</code>	Enters global configuration mode.
<b>Step 2</b>	<b>pce</b> <b>Example:</b> <code>RP/0/RSP0/CPU0:router(config)# pce</code>	Enables PCE and enters PCE configuration mode.
<b>Step 3</b>	<b>address ipv4 address</b> <b>Example:</b> <code>RP/0/RSP0/CPU0:router(config-pce)# address ipv4 192.168.0.1</code>	Configures a PCE IPv4 address.
<b>Step 4</b>	<b>state-sync ipv4 address</b> <b>Example:</b> <code>RP/0/RSP0/CPU0:router(config-pce)# state-sync ipv4 192.168.0.3</code>	Configures the remote peer for state synchronization.
<b>Step 5</b>	<b>tcp-buffer size size</b> <b>Example:</b> <code>RP/0/RSP0/CPU0:router(config-pce)# tcp-buffer size 1024000</code>	Configures the transmit and receive TCP buffer size for each PCEP session, in bytes. The default buffer size is 256000. The valid range is from 204800 to 1024000.
<b>Step 6</b>	<b>password {clear   encrypted} password</b> <b>Example:</b> <code>RP/0/RSP0/CPU0:router(config-pce)# password encrypted pwd1</code>	<p>Enables TCP MD5 authentication for all PCEP peers. Any TCP segment coming from the PCC that does not contain a MAC matching the configured password will be rejected. Specify if the password is encrypted or clear text.</p> <p><b>Note</b> TCP-AO and TCP MD5 are never permitted to be used simultaneously.</p>

	Command or Action	Purpose
<b>Step 7</b>	<b>tcp-ao</b> <i>key-chain</i> [ <b>include-tcp-options</b> ] [ <b>accept-ao-mismatch-connection</b> ] <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce)# tcp-ao pce_tcp_ao include-tcp-options</pre>	<p>Enables TCP Authentication Option (TCP-AO) authentication for all PCEP peers. Any TCP segment coming from the PCC that does not contain a MAC matching the configured key chain will be rejected.</p> <ul style="list-style-type: none"> <li>• <b>include-tcp-options</b>—Includes other TCP options in the header for MAC calculation.</li> <li>• <b>accept-ao-mismatch-connection</b>—Accepts connection even if there is a mismatch of AO options between peers.</li> </ul> <p><b>Note</b> TCP-AO and TCP MD5 are never permitted to be used simultaneously.</p>
<b>Step 8</b>	<b>segment-routing</b> { <b>strict-sid-only</b>   <b>te-latency</b> } <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce)# segment-routing strict-sid-only</pre>	<p>Configures the segment routing algorithm to use strict SID or TE latency.</p> <p><b>Note</b> This setting is global and applies to all LSPs that request a path from this controller.</p>
<b>Step 9</b>	<b>timers</b> <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce)# timers</pre>	<p>Enters timer configuration mode.</p>
<b>Step 10</b>	<b>keepalive</b> <i>time</i> <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce-timers)# keepalive 60</pre>	<p>Configures the timer value for locally generated keep-alive messages. The default time is 30 seconds.</p>
<b>Step 11</b>	<b>minimum-peer-keepalive</b> <i>time</i> <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce-timers)# minimum-peer-keepalive 30</pre>	<p>Configures the minimum acceptable keep-alive timer that the remote peer may propose in the PCEP OPEN message during session establishment. The default time is 20 seconds.</p>
<b>Step 12</b>	<b>reoptimization</b> <i>time</i> <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce-timers)# reoptimization 600</pre>	<p>Configures the re-optimization timer. The default timer is 1800 seconds.</p>

	Command or Action	Purpose
<b>Step 13</b>	<b>exit</b>  <b>Example:</b>  RP/0/RSP0/CPU0:router(config-pce-timers) # <b>exit</b>	Exits timer configuration mode and returns to PCE configuration mode.

## Configure the Disjoint Policy (Optional)

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

### SUMMARY STEPS

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

### DETAILED STEPS

#### Procedure

	Command or Action	Purpose
<b>Step 1</b>	<b>disjoint-path</b>  <b>Example:</b>  RP/0/RSP0/CPU0:router(config-pce) # <b>disjoint-path</b>	Enters disjoint configuration mode.
<b>Step 2</b>	<b>group-id</b> <i>value</i> <b>type</b> { <b>link</b>   <b>node</b>   <b>srlg</b>   <b>srlg-node</b> } [ <b>sub-id</b> <i>value</i> ]  <b>Example:</b>  RP/0/RSP0/CPU0:router(config-pce-disjoint) # <b>group-id 1 type node sub-id 1</b>	Configures the disjoint group ID and defines the preferred level of disjointness (the type of resources that should not be shared by the two paths): <ul style="list-style-type: none"> <li>• <b>link</b>—Specifies that links are not shared on the computed paths.</li> <li>• <b>node</b>—Specifies that nodes are not shared on the computed paths.</li> <li>• <b>srlg</b>—Specifies that links with the same SRLG value are not shared on the computed paths.</li> <li>• <b>srlg-node</b>—Specifies that SRLG and nodes are not shared on the computed paths.</li> </ul>

	Command or Action	Purpose
		<p>If a pair of paths that meet the requested disjointness level cannot be found, then the paths will automatically fallback to a lower level:</p> <ul style="list-style-type: none"> <li>• If the requested disjointness level is SRLG or node, then link-disjoint paths will be computed.</li> <li>• If the requested disjointness level was link, or if the first fallback from SRLG or node disjointness failed, then the lists of segments encoding two shortest paths, without any disjointness constraint, will be computed.</li> </ul>
<b>Step 3</b>	<b>strict</b> <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce-disjoint)# strict</pre>	(Optional) Prevents the automatic fallback behavior of the preferred level of disjointness. If a pair of paths that meet the requested disjointness level cannot be found, the disjoint calculation terminates and no new path is provided. The existing path is not modified.
<b>Step 4</b>	<b>lsp {1   2} pcc ipv4 address lsp-name lsp_name [shortest-path]</b> <b>Example:</b> <pre>RP/0/RSP0/CPU0:router(config-pce-disjoint)# lsp 1   pcc ipv4 192.168.0.1 lsp-name rtrA_t1   shortest-path RP/0/RSP0/CPU0:router(config-pce-disjoint)# lsp 2   pcc ipv4 192.168.0.5 lsp-name rtrE_t2</pre>	<p>Adds LSPs to the disjoint group.</p> <p>The <b>shortest-path</b> keyword forces one of the disjoint paths to follow the shortest path from the source to the destination. This option can only be applied to the the first LSP specified.</p>

## Global Maximum-delay Constraint

This feature allows a PCE to compare the cumulative latency of a computed path against a global maximum-delay constraint value. If the latency of the computed path exceeds this global constraint, the path is not considered valid. This ensures that all latency-based paths computed by the PCE and signaled to the PCCs in the network do not exceed this maximum-delay constraint.

```
pce
  constraints
    bounds
      cumulative
      type
      latency <1-4294967295> Bound metric value in microseconds
```

### Configuration

To configure a PCE for specifying maximum cumulative latency metric, you must complete the following configurations:

```
RP/0/RSP0/CPU0:ios(config)# pce
RP/0/RSP0/CPU0:ios(config-pce)# constraints
RP/0/RSP0/CPU0:ios(config-pce-constr)# bounds
```

```
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds)# cumulative
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds-type)# type latency 1000000
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds-type)#
```

### Verification

Verify using the **show** command:

```
RP/0/RSP0/CPU0:ios(config-pce-constr-bounds-type)# show
Wed Oct 12 22:18:22.962 UTC
pce
 constraints
  bounds
    cumulative
    type latency 1000000
  !
!
!
```

## PCE override of SR-TE policies

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

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

- Head-end – Where the SR-TE policy is instantiated
- Color – An unsigned non-zero 32-bit integer value that associates the SR Policy with an intent or objective (e.g., low latency).
- End-point – The destination of the SR-TE policy.

There are fewer PCEs in the network compared to the PCCs. However, PCE versions may be newer than the PCC version because it is easy to update the lesser PCEs than the higher number of PCCs. This ensures PCE supports more features with the latest Internet Assigned Numbers Authority (IANA) code points than PCCs.

IOS-XR PCE implementation is based on the PCEP standards and the implementations of the PCEP vary leading to interoperability challenges. To overcome these challenges, you can configure PCE with the override-rules (OVR). These OVRs have a filter part and an overriding part. The matching criteria filter matches all the (Segment Routing) SR policies against these OVRs, suppose they match, then the overriding part applies to these SR policies attributes.

For example, suppose that a PCC is outdated and does not support reporting the latency metric type. In that case, PCC reports to the LSP with the metric-type that it supports and the PCE applies the OVRs that you configure to update the metric-type upon reception.

## Usage guidelines

Some usage guidelines that apply to the PCE override of SR-TE policies feature.



- Lower sequence numbers have higher precedence. LSP can be matched to one override sequence at a time. It will be matched to the highest preference sequence and further matching stops.
- Show commands such as “show pce lsp” display values received from PCC, so applied changes are visible only in values, which are based on path-computation.
- In a PCE High Availability case, where there are multiple PCEs for a particular PCC, configure this feature likewise on all PCEs, to allow seamless transition of delegation between the PCEs.
- PCE (North Bound) NB-API clients only see overridden attributes in their communication with PCEs.
- The PCE OVR feature is seamless for PCCs. This means that PCCs will not know if the tunnel’s attributes are overridden. PCEs uses overridden attributes internally and sends only the original attributes to the PCCs.

## Limitations

- Supports only Segment Routing (SR) setup type.
- Supports only ten override rules.
- Only one color or color range per override rule is supported.
- Does not support SRv6 policies.
- You cannot configure the filtration option "all" under peer and LSP with another filtration option under the same config submode. For example, you cannot provide all and regex together under the LSP matching criteria.
- Only IPv4 Access Control Lists (ACL)s are supported for matching PCEP peers.
  - If the ACL is empty or there are no entries within the ACL, then it is implicit permit.
  - If the ACL is not configured but the name is specified, then it is implicit permit.
  - In all other cases, if no entries are matched in the ACL, then it is implicit deny.
  - The behavior aligns with ACL behavior in other areas on an XR router.

## Prerequisites

Before you configure the PCE override of SR-TE policies, you can use the **try-regex** utility to test the regexes. This utility is an optional step and does not alter the system.

This example shows how to verify if the

**override-rules sequence** *sequence-number* [ **matching-criteria** **lsp** { **all** | **name** *lsp-name-regex* } ] works:

```
Router# pce try-regex ^cp_c_[0-9]+$ cp_c_5000
Regex verification utility
Regex:      ^cp_c_[0-9]+$
Test string: cp_c_5000
Result:     Matched
```

## PCE override of PCC-initiated SR-TE policies

The Path Computation Element (PCE) is a key component of Software-Defined Networking (SDN) for Traffic Engineering (TE), which can dynamically compute and recompute paths for Segment Routing Traffic Engineering (SR-TE) policies. These computations are guided by specified path optimization metrics and constraints. By leveraging the PCE override feature, the PCE can modify certain attributes or constraints of these policies, enabling support for a broader range of use cases and enhanced functionality.

**Table 3: Feature History Table**

Feature Name	Release Information	Feature Description
PCE override of PCC-initiated policies	Release 7.7.1	<p>You can now override the LSP attributes in the PCEs based on the matching criteria of the path computation element protocol (PCEP) peer and LSP attributes. This feature improves the path computation of the PCC-initiated SR-TE policies.</p> <p>You can configure the PCE override rule for one, many, or all PCCs. This feature introduces the following commands:</p> <ul style="list-style-type: none"> <li>• <a href="#">override-rules (PCE)</a></li> <li>• <a href="#">pce try-regex</a></li> <li>• <a href="#">show pce override-rules detail</a></li> </ul>

Matching of SR-TE policies is done before PCInitiate is sent to PCC to allow modification of data included in messages sent to PCC before LSP is created.

The LSP matching is not re-evaluated again after receiving the next PCEP report message. Based on these matching criteria, you can override the values of the LSP attributes.

## Configure PCE override of PCC-initiated SR-TE policies

This configuration defines an **override rule** for a PCE on a network router, which allows the PCE to modify certain attributes of Label Switched Paths (LSPs) that meet specific criteria for PCC-initiated SR-TE policies.

### Procedure

#### Step 1 **configure**

##### Example:

```
Router# configure
```

Enters the global configuration mode.

#### Step 2 **pce**

##### Example:

```
Router(config)# pce
```

Enters the PCE configuration mode.

**Step 3**      **override-rules****Example:**

```
Router(config-pce)# override-rules
```

Enters the override rules submode.

**Step 4**      **sequence** *value***Example:**

```
Router(config-pce-ovr-rule)# sequence 100
```

Creates a rule with the sequence number 100.

**Step 5**      **matching-criteria****Example:**

```
Router(config-pce-ovr-rule-seq)# matching-criteria
```

Specifies a matching criteria. For example, **peer all**, **lsp all**, and so on.

**Step 6**      **peer all****Example:**

```
Router(config-pce-ovr-rule-crit)# peer  
Router(config-pce-ovr-rule-peer)# all  
Router(config-pce-ovr-rule-peer)# exit
```

Specifies that this rule applies to all peers.

.

**Step 7**      **lsp all** *range***Example:**

```
Router(config-pce-ovr-rule-crit)# lsp all  
Router(config-pce-ovr-rule-crit)# exit
```

Specifies all LSPs in the PCE tunnel database.

**Step 8**      **colors** *range***Example:**

```
Router(config-pce-ovr-rule-crit)# colors 0-50  
Router(config-pce-ovr-rule-crit)# exit
```

Defines that the colors in the 0 to 50 range to match this rule.

**Step 9**      **override metric type** *metric-type***Example:**

```
Router(config-pce-ovr-rule-seq)# override metric type igp
```

Specifies that the IGP (Interior Gateway Protocol) metric for matching LSPs is overridden by the PCE.

**Step 10**      **override constraints bandwidth** *value***Example:**

```
Router(config-pce-ovr-rule-seq)# override constraints bandwidth 1000
```

Specifies that the bandwidth constraint for matching LSPs is overridden and set to 1000.

**Step 11**      **commit****Example:**

```
Router(config-pce-ovr-rule-seq) # commit
```

Saves and apply these configuration changes.

**Running configuration**

These are the running configurations for the PCE override for the PCC-initiated SR-TE policies.

- Match all LSPs from all peers and modify metric type to latency:

```
pce
  override-rules
    sequence 1
      matching-criteria
        peer
          all
        lsp
          all
      !
    override
      metric
        type latency
  !
```

- Match LSP name using regex and peer based on provided ACL and change metric type to IGP:

```
pce
  override-rules
    sequence 1
      matching-criteria
        peer
          access-list ipv4 PCC1
        !
        lsp
          name ^cfg_test2_.*$
        !
      !
    override
      metric
        type igp
  !
```

- Match policy using specified Segment Routing (SR) policy color from all peers and change bandwidth to 1000 kbps:

```
pce
  override-rules
    sequence 1
      matching-criteria
        peer
          all
        !
        lsp
          colors 10
        !
      !
    override
```

```
constraints
bandwidth 1000
!
```

## Verification

Show command displays operational values, which are applied during the override. For example, a bandwidth value is converted into IEEE float format. The value that is displayed in the show command may not be equal to the value configured, because of the limited precision of IEEE format used in PCEP.

Verify using the show command **show pce override-rules [sequence <sequence>]**

This is the show command output with two override rules that are configured and each matched one LSP are:

```
Router# show pce override-rules details

PCE's Override Rule database:
-----
Sequence number: 1
Matching criteria:
  Peer:
    IPv4 ACL name: PCC1
    LSP
    Regex: ^cfg_test1_.*$
Override:
  Metric type: Latency
  Constraints:
Matching LSPs:
  Peer: 192.168.0.1, Tunnel name: cfg_test1_xxx_discr_5

Sequence number: 2
Matching criteria:
  Peer:
    IPv4 ACL name: PCC1
    LSP
    Regex: ^cfg_test2_.*$
Override:
  Metric type: IGP
  Constraints:
Matching LSPs:
  Peer: 192.168.0.1, Tunnel name: cfg_test2_xxx_discr_5
```

## PCE-Initiated SR Policies

Use cases based on centralized optimization, such as congestion mitigation solutions, rely on the ability of the PCE to signal and instantiate SR-TE policies in the network. We refer to this as PCE-initiated SR-TE policies.

PCE-initiated SR-TE policies can be triggered via Crossworks Network Controller (recommended approach) or via CLI at the PCE.

For more information on configuring SR-TE policies, see the [SR-TE Policy Overview](#).

The PCE deploys the SR-TE policy using PCC-PCE communication protocol (PCEP).

1. PCE sends a PCInitiate message to the PCC.
2. If the PCInitiate message is valid, the PCC sends a PCRpt message; otherwise, it sends PCErr message.
3. If the PCInitiate message is accepted, the PCE updates the SR-TE policy by sending PCUpd message.

You can achieve high-availability by configuring multiple PCEs with SR-TE policies. If the head-end (PCC) loses connectivity with one PCE, another PCE can assume control of the SR-TE policy.

### Configuration Example: PCE-Initiated SR Policy with Explicit SID List

To configure a PCE-initiated SR-TE policy, you must complete the following configurations:

1. Enter PCE configuration mode.
2. Create the segment list.



**Note** When configuring an explicit path using IP addresses of intermediate links, the SR-TE process prefers the protected Adj-SID of the link, if one is available.

3. Create the policy.

```
/* Enter PCE configuration mode and create the SR-TE segment lists */
Router# configure
Router(config)# pce

/* Create the SR-TE segment lists */
Router(config-pce)# segment-routing
Router(config-pce-sr)# traffic-eng
Router(config-pce-sr-te)# segment-list name addr2a
Router(config-pce-sr-te-sl)# index 10 address ipv4 10.1.1.2
Router(config-pce-sr-te-sl)# index 20 address ipv4 10.2.3.2
Router(config-pce-sr-te-sl)# index 30 address ipv4 10.1.1.4
Router(config-pce-sr-te-sl)# exit

/* Create the SR-TE policy */
Router(config-pce-sr-te)# peer ipv4 10.1.1.1
Router(config-pce-sr-te)# policy P1
Router(config-pce-sr-te-policy)# color 2 end-point ipv4 2.2.2.2
Router(config-pce-sr-te-policy)# candidate-paths
Router(config-pce-sr-te-policy-path)# preference 50
Router(config-pce-sr-te-policy-path-preference)# explicit segment-list addr2a
Router(config-pce-sr-te-policy-path-preference)# commit
Router(config-pce-sr-te-policy-path-preference)# end
Router(config)#
```

### Running Config

```
pce
segment-routing
traffic-eng
segment-list name addr2a
index 10 address ipv4 10.1.1.2
index 20 address ipv4 10.2.3.2
index 30 address ipv4 10.1.1.4
!
peer ipv4 10.1.1.1
policy P1
color 2 end-point ipv4 2.2.2.2
candidate-paths
preference 50
```

```

explicit segment-list addr2a
!
!

```

## SR-PCE Flexible Algorithm Multi-Domain Path Computation

**Table 4: Feature History Table**

Feature Name	Release Information	Feature Description
SR-PCE Flexible Algorithm Multi-Domain Path Computation	Release 7.3.1	With this feature, SR-PCE can use Flexible Algorithms to compute multi-domain paths.

Flexible Algorithm provides a traffic engineered path automatically computed by the IGP to any destination reachable by the IGP. With the SR-PCE Flexible Algorithm Multi-Domain Path Computation feature, SR-PCE can use Flexible Algorithms to compute multi-domain paths. See the [Enabling Segment Routing Flexible Algorithm](#) chapter for information about Segment Routing Flexible Algorithm.

The SR-PCE Flexible Algorithm Multi-Domain Path Computation feature incorporates the following functionality:

- BGP-LS has been augmented to allow selected nodes to advertise the Flexible Algorithm definition (FAD) to the SR-PCE
- PCEP has been augmented (vendor-specific object) to allow a PCC to indicate SR policy constraint based on the Flexible Algorithm instance number
- SR-PCE algorithms have been augmented to compute paths based on a Flexible Algorithm constraint

The SR-PCE Flexible Algorithm multi-domain path computation requires the following:

- The same Flexible Algorithm instance ID is used across domains.
- The metric for those Flexible Algorithm instances must be the same across domains.
- The affinity constraints for those Flexible Algorithm instances may be different across domains.
- Multiple Flexible Algorithms can exist in a domain.

For example, considering a multi-domain topology (Domain 1 and Domain 2), the following scenarios meet the requirements listed above:

Scenario	Domain 1	Domain 2
Scenario 1	Flexible Algorithm 128, metric delay	Flexible Algorithm 128, metric delay
Scenario 2	Flexible Algorithm 128, metric delay	Flexible Algorithm 128, metric delay, exclude affinity blue
Scenario 3	Flexible Algorithm 128, metric delay, exclude affinity yellow	Flexible Algorithm 128, metric delay, exclude affinity blue

Scenario	Domain 1	Domain 2
Scenario 4	Flexible Algorithm 128, metric delay	Flexible Algorithm 128, metric delay
	Flexible Algorithm 129, metric IGP	Flexible Algorithm 129, metric IGP

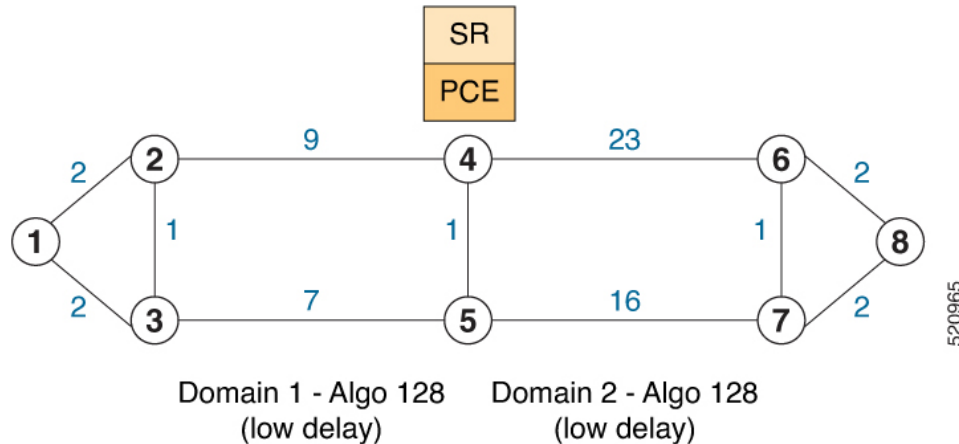


**Note** The use of a Flexible Algorithm constraint in a multi-domain SR topology does not preclude the use of an SR policy that are optimized for a particular metric type. For example, a policy can request a PCE for a Multi Domain policy based on metric delay. SR-PCE computes the path and encodes it with regular prefix SIDs and Adj-SIDs as required. Alternatively, a policy can request to have a constraint for a Flexible Algorithm instance X, which is defined in multiple domains and it minimizes based on metric delay. In this case, the SR-PCE computes the multi-domain path and encodes it using only Flexible Algorithm prefix SIDs. This case benefits from the optimized label stack size that Flexible Algorithm provides (1 label per domain).

## Example: SR-PCE Flexible Algorithm Multi-Domain Path Computation Use Case

The following use case depicts a multi-domain topology with two IS-IS processes, each with a Flexible Algorithm instance of 128 that minimizes metric delay. A multi-domain SR policy programmed at Node 1 leverages a Flexible Algorithm 128 path computed by the SR-PCE toward Node 8.

**Figure 1: Multi-Domain Topology**



### Configuration on Node 8

#### IS-IS and Flexible Algorithm Configuration

```
router isis 2
 is-type level-2-only
 net 49.0002.0000.0000.0008.00
 distribute link-state
 flex-algo 128
   metric-type delay
   advertise-definition
```



```

address-family ipv4 unicast
  metric-style wide
  router-id 10.1.1.8
  segment-routing mpls
!
interface Loopback0
  passive
  address-family ipv4 unicast
  prefix-sid absolute 16008
  prefix-sid algorithm 128 absolute 16808
!

```

## Configuration on Node 4 (ABR/ASBR)

### IS-IS and Flexible Algorithm Configuration

```

router isis 1
  is-type level-2-only
  net 49.0001.0000.0000.0004.00
  distribute link-state instance-id 100
  flex-algo 128
    metric-type delay
    advertise-definition

  address-family ipv4 unicast
    metric-style wide
    router-id 10.1.1.4
    segment-routing mpls
  !
  interface Loopback0
    passive
    address-family ipv4 unicast
    prefix-sid absolute 16004
    prefix-sid algorithm 128 absolute 16804
  !
router isis 2
  is-type level-2-only
  net 49.0002.0000.0000.0004.00
  distribute link-state instance-id 200
  flex-algo 128
    metric-type delay
    advertise-definition

  address-family ipv4 unicast
    metric-style wide
    router-id 10.1.1.4
    segment-routing mpls
  !
  interface Loopback0
    passive
    address-family ipv4 unicast
    prefix-sid absolute 16004
    prefix-sid algorithm 128 absolute 16804
  !

```

### BGP-LS Configuration

```

router bgp 65000
  bgp router-id 10.1.1.4
  address-family link-state link-state
  !
  neighbor-group AS65000-LS-group

```

```

remote-as 65000
update-source Loopback0
address-family link-state link-state
!
!
neighbor 10.1.1.10
use neighbor-group AS65000-LS-group
description *** To SR-PCE ***
!
!
!
```

## Configuration on Node 1

### IS-IS and Flexible Algorithm Configuration

```

router isis 1
is-type level-2-only
net 49.0001.0000.0000.0001.00
distribute link-state
flex-algo 128
    metric-type delay
    advertise-definition

address-family ipv4 unicast
    metric-style wide
    router-id 10.1.1.1
    segment-routing mpls
!
interface Loopback0
    passive
    address-family ipv4 unicast
        prefix-sid absolute 16001
        prefix-sid algorithm 128 absolute 16801
!
```

### SR Policy Configuration

```

segment-routing
traffic-eng
    policy FOO
        color 100 end-point ipv4 10.1.1.8
        candidate-paths
            preference 100
            dynamic
                pcep
                !
            !
        constraints
            segments
                sid-algorithm 128
            !
        !
    !
!
!
!
```

### PCC Configuration

```

segment-routing
traffic-eng
```

```

pcc
 source-address ipv4 10.1.1.1
 pce address ipv4 10.1.1.10
 precedence 10
 !
 report-all
 !
 !
 !

```

### Configuration on PCE

```

pce
 address ipv4 10.1.1.10
 rest
 !
 !
router bgp 65000
 bgp router-id 10.1.1.10
 address-family link-state link-state
 !
 neighbor-group AS65000-LS-group
  remote-as 65000
  update-source Loopback0
  address-family link-state link-state
  !
 !
 neighbor 10.1.1.4
  use neighbor-group AS65000-LS-group
  description *** To Node-4 ***
  !
 !
 neighbor 10.1.1.5
  use neighbor-group AS65000-LS-group
  description *** To Node-5 ***
  !
 !
 !

```

## ACL Support for PCEP Connection

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

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

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

To apply an ACL to the PCE, use the **pce peer-filter ipv4 access-list *acl\_name*** command.

The following example shows how to configure an ACL and apply it to the PCE:

```

pce
 address ipv4 10.1.1.5
 peer-filter ipv4 access-list sample-peer-filter
!
ipv4 access-list sample-peer-filter
10 permit ipv4 host 10.1.1.6 any
20 permit ipv4 host 10.1.1.7 any
30 deny ipv4 any any
!

```

## Anycast SID-Aware Path Computation

This feature allows the SR-TE head-end or SR-PCE to compute a path that is encoded using Anycast prefix SIDs of nodes along the path.

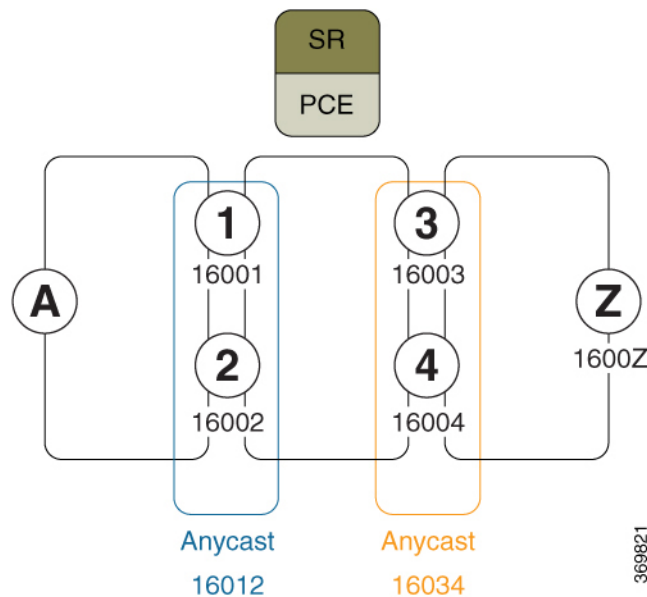
An Anycast SID is a type of prefix SID that identifies a set of nodes and is configured with n-flag clear. The set of nodes (Anycast group) is configured to advertise a shared prefix address and prefix SID. Anycast routing enables the steering of traffic toward multiple advertising nodes, providing load-balancing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



**Note** For information on configuring Anycast SID, see [Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface](#) and [Configuring a Prefix-SID on the OSPF-Enabled Loopback Interface](#).

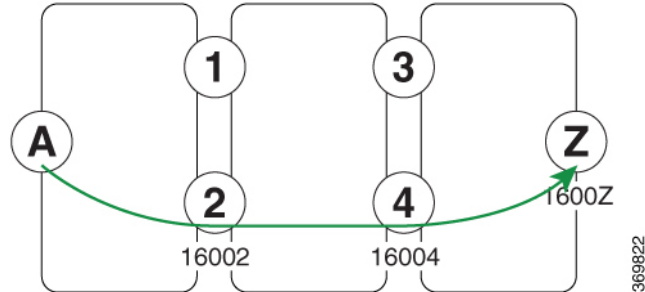
This example shows how Anycast SIDs are inserted into a computed SID list.

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 nodes A and Z are provider edge (PE) routers in the same VPN. Node A receives a VPN route with BGP next-hop to node Z. Node A resolves the SR path to node Z based on ODN behaviors with delegation of path computation to SR-PCE.

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



Assume that the computed path from node A to node Z traverses node 2 and node 4. This translates to SID list {16002, 16004, 1600Z} when node SIDs are leveraged to encode the path.

When an Anycast SID-aware path is requested, the path computation algorithm performs the following:

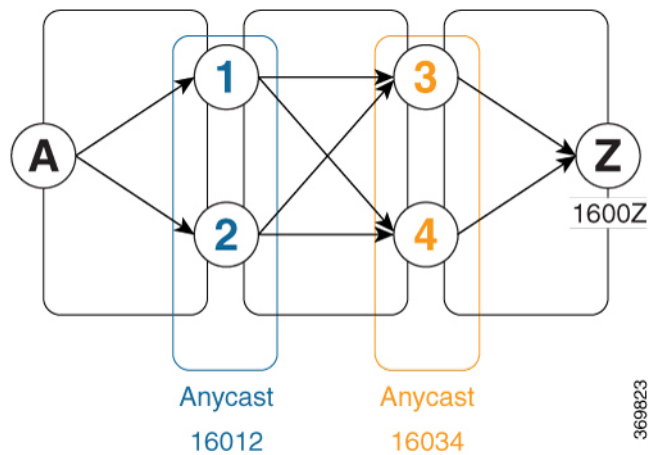
- **Path Computation**—Computes the path according to optimization objectives and constraints
- **Path Encoding**—Encodes the path in a SID list leveraging node-SIDs and adj-SIDs as applicable
- **Anycast SID Replacement**—Reiterates the original SID list by replacing node SIDs with Anycast SIDs present on the nodes along the computed path.

If a node has multiple Anycast SIDs, the algorithm considers them according to their weights. See [Weighted Anycast SIDs, on page 24](#).

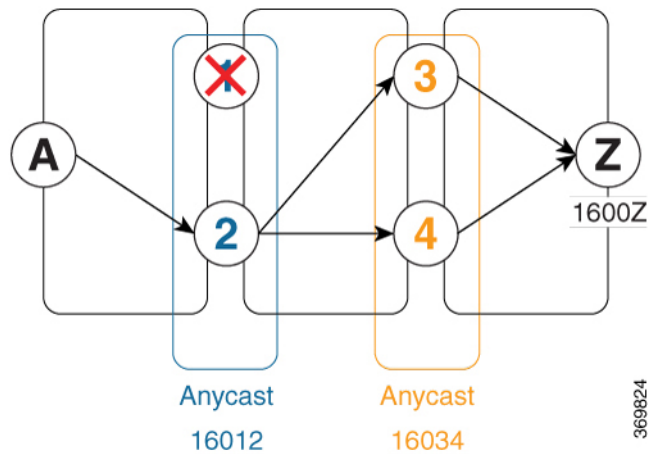
- **Optimality Validation**—The new paths are validated against the original optimization objectives and constraints (maintain same cumulative metric as original SID list and do not violate path constraints).
- **Anycast SID Promotion**—If the optimality validation is successful, then the Anycast-encoded SID list is signaled and instantiated in the forwarding.

The following figure depicts cumulative metrics between nodes in the network.

Under these conditions, the optimality check is met, and therefore, the Anycast-encoded SID list from node A to node Z is {16012, 16034, 1600Z}.



The Anycast SID aware path computation also provides resiliency. For example, if one of the ABRs (in this case, ABR 1) becomes unavailable or unreachable, the path from node A to node Z {16012,16034,1600Z} will still be valid and usable.



### Configuration Examples

1. Configure Prefix SIDs on the ABR nodes.
  - a. Configure each node with a node SID.
  - b. Configure each group of nodes with a shared Anycast SID.

See [Configuring a Prefix-SID on the IS-IS Enabled Loopback Interface](#) 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 request Anycast SID-aware path computation for the following SR policy types:

- Local SR policy with PCC-initiated path computation at the head-end router:

```
segment-routing  
  traffic-eng  
    policy FOO  
      color 10 end-point ipv4 10.1.1.10  
      candidate-paths  
        preference 100  
        dynamic  
        anycast-sid-inclusion
```

- Local SR policy with PCC-initiated/PCE-delegated path computation at the SR-PCE:

```
segment-routing  
  traffic-eng  
    policy BAR  
      color 20 end-point ipv4 10.1.1.20  
      candidate-paths  
        preference 100  
        dynamic  
        pcep  
        anycast-sid-inclusion
```

- On-demand SR policies with a locally computed dynamic path at the head-end, or centrally computed dynamic path at the SR-PCE:

```
segment-routing  
  traffic-eng  
    on-demand color 10  
    dynamic  
    anycast-sid-inclusion
```

- On-demand SR policies with centrally computed dynamic path at the SR-PCE:

```
segment-routing  
  traffic-eng  
    on-demand color 20  
    dynamic  
    pcep  
    anycast-sid-inclusion
```

## Weighted Anycast SIDs

**Table 5: Feature History Table**

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

Weighted Anycast nodes advertise a cost along with the Anycast SID. This cost serves as a weight. The native SR path computation algorithms are augmented to compute optimum paths relying on Weighted Anycast SIDs during path encoding. Traffic to the SID is then distributed according to the weights.

The following example shows how node SID, Anycast SID, and Weighted Anycast SID are applied on node 1:

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



# SR-PCE IPv4 Unnumbered Interface Support

This feature allows IPv4 unnumbered interfaces to be part of an SR-PCE topology database.

An unnumbered IPv4 interface is not identified by its own unique IPv4 address. Instead, it is identified by the router ID of the node where this interfaces resides and the local SNMP index assigned for this interface.

This feature provides enhancements to the following components:

- IGPs (IS-IS and OSPF):
  - Support the IPv4 unnumbered interfaces in the SR-TE context by flooding the necessary interface information in the topology
- SR-PCE:



## Note

SR-PCE and path computation clients (PCCs) need to be running Cisco IOS XR 7.0.2 or later.

- Compute and return paths from a topology containing IPv4 unnumbered interfaces.
- Process reported SR policies from a head-end router that contain hops with IPv4 unnumbered adjacencies.

PCEP extensions for IPv4 unnumbered interfaces adhere to IETF RFC8664 “PCEP Extensions for Segment Routing” (<https://datatracker.ietf.org/doc/rfc8664/>). The unnumbered hops use a Node or Adjacency Identifier (NAI) of type 5. This indicates that the segment in the explicit routing object (ERO) is an unnumbered adjacency with an IPv4 ID and an interface index.

- SR-TE process at the head-end router:
  - Compute its own local path over a topology, including unnumbered interfaces.
  - Process PCE-computed paths that contain hops with IPv4 unnumbered interfaces.
  - Report a path that contains hops with IPv4 unnumbered interfaces to the PCE.

## Configuration Example

The following example shows how to configure an IPv4 unnumbered interface:

```
RP/0/0/CPU0:rtrA(config)# interface GigabitEthernet0/0/0/0
RP/0/0/CPU0:rtrA(config-if)# ipv4 point-to-point
RP/0/0/CPU0:rtrA(config-if)# ipv4 unnumbered Loopback0
```

To bring up the IPv4 unnumbered adjacency under the IGP, configure the link as point-to-point under the IGP configuration. The following example shows how to configure the link as point-to-point under the IGP configuration:

```
RP/0/0/CPU0:rtrA(config)# router ospf one
RP/0/0/CPU0:rtrA(config-ospf)# area 0
RP/0/0/CPU0:rtrA(config-ospf-ar)# interface GigabitEthernet0/0/0/0
```

```
RP/0/0/CPU0:rtrA(config-ospf-ar-if)# network point-to-point
```

## Verification

Use the **show ipv4 interface** command to display information about the interface:

```
RP/0/0/CPU0:rtrA# show ipv4 interface GigabitEthernet0/0/0/0 brief
Tue Apr  2 12:59:53.140 EDT
Interface                               IP-Address      Status          Protocol
GigabitEthernet0/0/0/0                 192.168.0.1     Up              Up
```

This interface shows the IPv4 address of Loopback0.

Use the **show snmp interface** command to find the SNMP index for this interface:

```
RP/0/0/CPU0:rtrA# show snmp interface
Tue Apr  2 13:02:49.190 EDT
ifName : Null0                ifIndex : 3
ifName : Loopback0            ifIndex : 10
ifName : GigabitEthernet0/0/0/0 ifIndex : 6
```

The interface is identified with the pair (IPv4:192.168.0.1, index:6).

Use the **show ospf neighbor** command to display the adjacency:

```
RP/0/0/CPU0:rtrA# show ospf neighbor gigabitEthernet 0/0/0/0 detail
...
Neighbor 192.168.0.4, interface address 192.168.0.4
  In the area 0 via interface GigabitEthernet0/0/0/0
  Neighbor priority is 1, State is FULL, 6 state changes
...
Adjacency SIDs:
  Label: 24001,      Dynamic, Unprotected
Neighbor Interface ID: 4
```

The output of the **show pce ipv4 topology** command is enhanced to display the interface index instead of the IP address for unnumbered interfaces:

```
RP/0/0/CPU0:sr-pce# show pce ipv4 topology
...
Link[2]: unnumbered local index 6, remote index 4
Local node:
  OSPF router ID: 192.168.0.1 area ID: 0 ASN: 0
Remote node:
  TE router ID: 192.168.0.4
  OSPF router ID: 192.168.0.4 area ID: 0 ASN: 0
Metric: IGP 1, TE 1, Latency 1 microseconds
Bandwidth: Total 125000000 Bps, Reservable 0 Bps
Admin-groups: 0x00000000
Adj SID: 24001 (unprotected)
```

The output of **show pce lsp detail** command includes unnumbered hops:

```
RP/0/0/CPU0:sr-pce# show pce lsp detail
...
Reported path:
Metric type: TE, Accumulated Metric 3
SID[0]: Adj unnumbered, Label 24001, local 192.168.0.1(6), remote 192.168.0.4(4)
SID[1]: Adj unnumbered, Label 24002, local 192.168.0.4(7), remote 192.168.0.3(7)
SID[2]: Adj unnumbered, Label 24000, local 192.168.0.3(5), remote 192.168.0.2(5)
Computed path: (Local PCE)
Computed Time: Wed Apr 03 11:01:46 EDT 2019 (00:01:06 ago)
```

```

Metric type: TE, Accumulated Metric 3
SID[0]: Adj unnumbered, Label 24001, local 192.168.0.1(6), remote 192.168.0.4(4)
SID[1]: Adj unnumbered, Label 24002, local 192.168.0.4(7), remote 192.168.0.3(7)
SID[2]: Adj unnumbered, Label 24000, local 192.168.0.3(5), remote 192.168.0.2(5)

```

## Inter-Domain Path Computation Using Redistributed SID

Table 6: Feature History Table

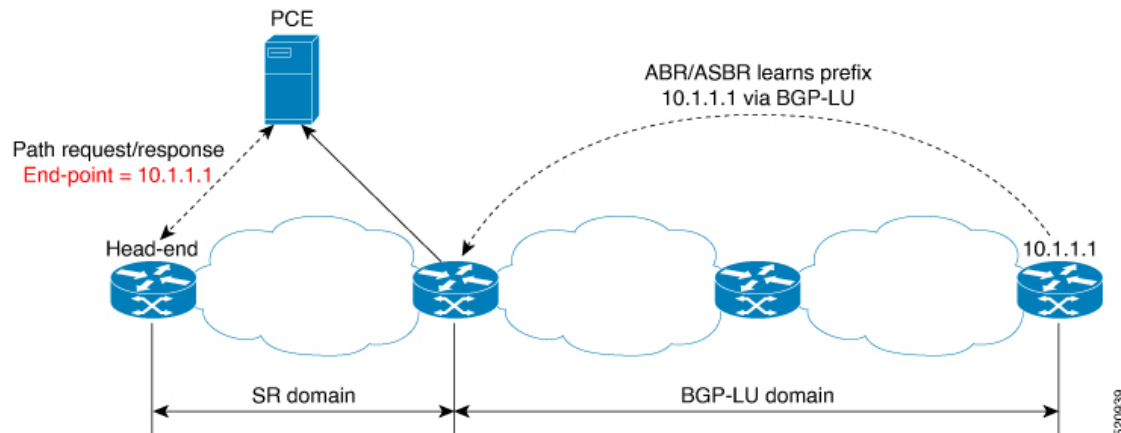
Feature Name	Release Information	Feature Description
SR-PCE: Inter-Domain Computation using Redistributed SID (SRTE - BGP-LU Domains Interworking)	Release 7.3.1	This feature adds new functionality to the SR-PCE that enables it to compute a path for remote non-SR end-point device distributed by BGP-LU.

A Path Computation Element (PCE) computes SR-TE paths based on SR topology database that stores connectivity, state, and TE attributes of SR network nodes and links. BGP Labeled Unicast (BGP-LU) provides MPLS transport across IGP boundaries by advertising loopbacks and label binding of impact edge and border routers across IGP boundaries.

This feature adds new functionality to the SR-PCE that enables it to compute a path for remote non-SR end-point device distributed by BGP-LU.

The remote end-point device in the BGP-LU domain is unknown to the SR-PCE. For the SR-PCE to know about the end-point device, the gateway ABR/ASBR learns the end-point prefix via BGP-LU. The prefix is then redistributed to SR-PCE topology database from the gateway ABR/ASBR. SR-PCE then can compute the best path from the head-end device to the selected gateway router.

The following topology shows an SR domain and a BGP-LU domain, with a gateway ABR/ASBR between the two domains.

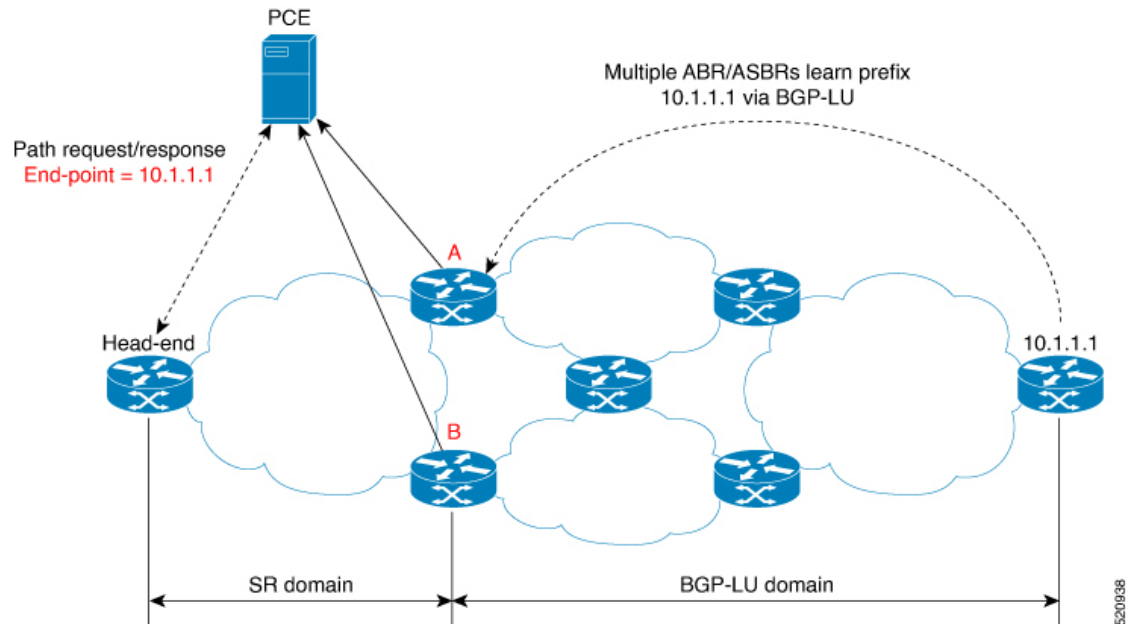


1. The gateway ABR/ASBR is configured with BGP/IGP helper to learn the remote prefix through BGP-LU and redistribute the remote prefix to the IGP helper, then to SR-PCE.
2. The SR-PCE selects the best gateway node to BGP-LU domain and computes the path to reach the remote prefix through the gateway node.

3. The head-end device in the SR domain requests a path to the remote destination and signals the SR profile interworking with the BGP-LU domain.

The BGP-LU prefix advertisement to SR-PCE Traffic Engineer Database (TED) is done by creating an IGP helper on the ABR/ASBR to redistribute BGP-LU prefix information to IGP. IGP then sends the prefix information to the SR-PCE via BGP-LS.

If there are multiple ABR/ASBRs advertising the same remote BGP-LU prefix, the SR-PCE selects the best gateway node to the BGP-LU domain using the accumulative metric from the head-end device to the gateway and the advertised metric from the gateway to the destination.



## Example: Inter-Domain Path Computation Using Redistributed SID

The following examples show the configurations for the IGP helper, BGP-LU, and proxy BGP-SR:

### Configuration on the End-Point Device

Configure the end-point device to allocate a label for the BGP-LU prefix on the end-point device:

```
router bgp 3107
  bgp router-id 1.0.0.8
  address-family ipv4 unicast
    network 1.0.0.8/32 route-policy bgplu-com
    allocate-label all

route-policy bgplu-com
  set community (65002:999)
end-policy
```

### Configuration on the Gateway ABR/ASBR

1. Configure the remote prefix set and create the route policy for the BGP-LU domain:

```

prefix-set bgplu
  1.0.0.7/32,
  1.0.0.8/32,
  1.0.0.101/32,
  1.0.0.102/32
end-set
!

route-policy bgp2isis
  if destination in bgplu then
    pass
  else
    drop
  endif
end-policy
!
end

```

2. Configure the helper IGP instance on the Loopback interface:

```

router isis 101
  is-type level-2-only
  net 49.0001.0000.1010.1010.00
  distribute link-state instance-id 9999
  nsf cisco
  nsf lifetime 120
  address-family ipv4 unicast
  metric-style wide
  maximum-paths 64
  router-id Loopback10
  redistribute bgp 3107 metric 200 route-policy bgp2isis
  segment-routing mpls sr-prefer
!
interface Loopback10 >>> this loopback is for gateway SR-TE node-id
  passive
  address-family ipv4 unicast
  prefix-sid index 2001 explicit-null

```

3. Configure the gateway proxy BGP-SR and SR Mapping Server to allocate SR labels:

```

router bgp 3107
  address-family ipv4 unicast
  segment-routing prefix-sid-map
  allocate-label all

segment-routing
global-block 16000 23999
mapping-server
  prefix-sid-map
  address-family ipv4
  1.0.0.7/32 2007
  1.0.0.8/32 2008
  1.0.0.101/32 2101
  1.0.0.102/32 2102

```

# Configuring the North-Bound API on SR-PCE

*Table 7: Feature History Table*

*Table 8: Feature History Table*

Feature Name	Release Information	Feature Description
SR-PCE: Stateful North-Bound API for Tree-SID	Release 7.5.1	<p>The SR-PCE provides a north-bound HTTP-based API to allow communication between the SR-PCE and the Cisco Crosswork Optimization Engine.</p> <p>This release adds stateful north-bound APIs to support real-time monitoring of Tree-SID states on the SR-PCE using a subscription model.</p> <p>For more information, refer to the <a href="#">Cisco Crosswork Optimization Engine User Guides</a>.</p>

*Table 9: Feature History Table*

Feature Name	Release Information	Feature Description
SR-PCE: North-Bound API for SRv6 and Flexible Algorithm in Cisco Optimization Engine (COE) v3.0 release	Release 7.3.2	<p>The SR-PCE provides a north-bound HTTP-based API to allow communication between the SR-PCE and the Cisco Crosswork Optimization Engine.</p> <p>This release adds support for the following:</p> <ul style="list-style-type: none"> <li>• Reporting of Flexible Algorithm participation and definitions</li> <li>• SRv6 topology information (nodes, links, Node uSIDs and Adj uSIDs)</li> <li>• SRv6 uSID list and uB6 SIDs allocated for a policy</li> </ul> <p>For more information, refer to the <a href="#">Cisco Crosswork Optimization Engine User Guides</a>.</p>

The SR-PCE provides a north-bound HTTP-based API to allow communication between SR-PCE and external clients and applications.

Over this API, an external application can leverage the SR-PCE for topology discovery, SR policy discovery, and SR policy instantiation.

The Cisco Crosswork Optimization Engine is an application that leverages the SR-PCE. For more information, refer to the [Cisco Crosswork Optimization Engine User Guides](#).

Use the following commands under PCE configuration mode to configure the API to allow communication between SR-PCE and external clients or applications.

Command	Description
<b>rest authentication basic</b>	(Optional) Specify basic (plaintext) authentication. By default, authentication is disabled.
<b>rest username</b> <i>password</i> { <b>clear</b>   <b>encrypted</b> } <i>password</i>	Add credentials when connecting to API.  <b>Note</b> This command is used only if authentication is configured.
<b>rest sibling ipv4</b> <i>address</i>	Opens a synchronization channel to another PCE in the same high availability (HA) pair.  <b>Note</b> For more information regarding SR-PCE HA pairs, refer to the <a href="#">Multiple Cisco SR-PCE HA Pairs</a> chapter of the <a href="#">Cisco Crosswork Optimization Engine 1.2.1 User Guide</a> .

Command	Description
<b>api authentication</b> { <b>basic</b>   <b>digest</b> }	Specify the type of authentication: <ul style="list-style-type: none"> <li>• <b>basic</b> – Use HTTP Basic authentication (plaintext)</li> <li>• <b>digest</b> – Use HTTP Digest authentication (MD5)</li> </ul>
<b>api username</b> <i>password</i> { <b>clear</b>   <b>encrypted</b> } <i>password</i>	Add credentials when connecting to API.
<b>api sibling ipv4</b> <i>address</i>	Opens a synchronization channel to another PCE in the same high availability (HA) pair.  <b>Note</b> For more information regarding SR-PCE HA pairs, refer to the <a href="#">Multiple Cisco SR-PCE HA Pairs</a> chapter of the <a href="#">Cisco Crosswork Optimization Engine 1.2.1 User Guide</a> .

**Example: Configuring API on SR-PCE**

```
pce
address ipv4 10.1.1.100
rest
  user admin
  password encrypted 1304131F0202
  !
  authentication basic
  sibling ipv4 10.1.1.200
  !
!
end
```

```
pce
address ipv4 10.1.1.100
api
  user admin
  password encrypted 1304131F0202
  !
  authentication digest
  sibling ipv4 10.1.1.200
  !
!
end
```

The following example shows the current active connections:

```
RP/0/0/CPU0:pce1# show tcp brief | i 8080
Thu Aug  6 00:40:15.408 PDT
0xe9806fb8 0x60000000      0      0  :::8080          :::0          LISTEN
0xe94023b8 0x60000000      0      0  10.1.1.100:50487  10.1.1.200:8080  ESTAB
0xeb20bb40 0x60000000      0      0  10.1.1.100:8080   10.1.1.200:44401  ESTAB
0xe98031a0 0x60000000      0      0  0.0.0.0:8080      0.0.0.0:0       LISTEN
```

The first and fourth entries show the API server listening for IPv4 and IPv6 connections.

The second and third entries show the established sibling connection between PCE1 (10.1.1.100) and PCE2 (10.1.1.200).