Segment Routing with OSPF

About OSPF

Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP) developed by the OSPF working group of the Internet Engineering Task Force (IETF). Designed expressly for IP networks, OSPF supports IP subnetting and tagging of externally derived routing information. OSPF also allows packet authentication and uses IP multicast when sending and receiving packets.

Segment routing configuration on the OSPF protocol can be applied at the process or the area level. If you configure segment routing at the process level, it is enabled for all the areas. However, you can enable ore disable it per area level.

Segment routing on the OSPF protocol supports the following:

- OSPFv2 control plane
- Multi-area
- IPv4 prefix SIDs for host prefixes on loopback interfaces
- Adjacency SIDs for adjacencies

Adjacency SID Advertisement

OSPF supports the advertisement of segment routing adjacency SID. An Adjacency Segment Identifier (Adj-SID) represents a router adjacency in Segment Routing.

A segment routing-capable router may allocate an Adj-SID for each of its adjacencies and an Adj-SID sub-TLV is defined to carry this SID in the Extended Opaque Link LSA.

OSPF allocates the adjacency SID for each OSPF neighbor if the OSPF adjacency which are in two way or in FULL state. OSPF allocates the adjacency SID only if the segment routing is enabled. The label for adjacency SID is dynamically allocated by the system. This eliminates the chances of misconfiguration, as this has got only the local significance.

Connected Prefix-SID

OSPFv2 supports the advertisement of prefix SID for address associated with the loopback interfaces. In order to achieve this, OSPF uses Extended Prefix Sub TLV in its opaque Extended prefix LSA. When OSPF receives this LSA from its neighbor, SR label is added to the RIB corresponding to received prefix based upon the information present in extended prefix sub TLV.

For configuration, segment-routing has to be enabled under OSPF and corresponding to loopback interface that is configured with OSPF, prefix-sid mapping is required under the segment routing module.

Note SID will only be advertised for loopback addresses and only for intra-area and inter-area prefix types. No SID value will be advertised for external or NSSA prefixes.

Prefix Propagation Between Areas

To provide segment routing support across the area boundary, OSPF is required to propagate SID values between areas. When OSPF advertises the prefix reachability between areas, it checks if the SID has been advertised for the prefix. In a typical case, the SID value come from the router, which contributes to the best path to the prefix in the source area. In this case, OSPF uses such SID and advertises it between the areas. If the SID value is not advertised by the router which contributes to the best path inside the area, OSPF will use the SID value coming from any other router inside the source area.

Segment Routing Global Range Changes

OSPF advertises it's segment routing capability in terms of advertising the SID/Label Range TLV. In OSPFv2, SID/Label Range TLV is a carried in Router Information LSA.

The segment routing global range configuration will be under the “segment-routing mpls” configuration. When the OSPF process comes, it will get the global range values from segment-routing and subsequent changes should be propagated to it.

When OSPF segment routing is configured, OSPF must request an interaction with the segment routing module before OSPF segment routing operational state can be enabled. If the SRGB range is not created, OSPF will not be enabled. When an SRGB change event occurs, OSPF makes the corresponding changes in it's sub-block entries.

Conflict Handling of SID Entries

In an ideal situation, each prefix should have unique SID entries assigned.

When there is a conflict between the SID entries and the associated prefix entries use any of the following methods to resolve the conflict:

- Multiple SIDs for a single prefix - If the same prefix is advertised by multiple sources with different SIDs, OSPF will install the unlabeled path for the prefix. The OSPF takes into consideration only those SIDs that are from reachable routers and ignores those from unreachable routers. When multiple SIDs are advertised for a prefix, which is considered as a conflict, no SID will be advertised to the attached-areas for the prefix. Similar logic will be used when propagating the inter-area prefixes between the backbone and the non-backbone areas.
- Out of Range SID - For SIDs that do not fit in our SID range, labels are not used while updating the RIB.

MPLS Forwarding on an Interface

MPLS forwarding must be enabled before segment routing can use an interface. OSPF is responsible for enabling MPLS forwarding on an interface.

When segment routing is enabled for a OSPF topology, or OSPF segment routing operational state is enabled, it enables MPLS for any interface on which the OSPF topology is active. Similarly, when segment routing is disabled for a OSPF topology, it disables the MPLS forwarding on all interfaces for that topology.

MPLS forwarding is not supported on an interface which terminates at the IPIP/GRE tunnel.

Configuring Segment Routing with OSPFv2

Configure segment routing with OSPFv2 protocol.

Before you begin

Confirm that the following conditions are met before configuring segment routing with OSPFv2:

- The OSPFv2 feature is enabled.
- The segment-routing feature is enabled.
- Segment routing is enabled under OSPF.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**]**router ospf** *process* **Example:** `switch(config)# router ospf test`	Enables the OSPF mode.
Step 3	**segment-routing** **Example:** `switch(config-router)# segment-routing mpls`	Configures the segment routing functionality under OSPF.

Configuring Segment Routing on OSPF Network- Area Level

Before you begin

Before you configure segment routing on OSPF network, OSPF must be enabled on your network.

Procedure

	Command or Action	Purpose
Step 1	**router ospf process** **Example:** `switch(config)# router ospf test`	Enables the OSPF mode.
Step 2	**area** <*area id*> **segment-routing** [**mpls** | **disable**] **Example:** `switch(config-router)# area 1 segment-routing mpls`	Configures segment routing mpls mode in a specific area.
Step 3	[**no**]**area** <*area id*> **segment-routing** [**mpls** | **disable**] **Example:** `switch(config-router)#area 1 segment-routing disable`	Disables segment routing mpls mode for the specified area.
Step 4	**show ip ospf** *process* **segment-routing** **Example:** `switch(config-router)# show ip ospf test segment-routing`	Shows the output for configuring segment routing under OSPF.

Configuring Prefix-SID for OSPF

This task explains how to configure prefix segment identifier (SID) index under each interface.

Before you begin

Segment routing must be enabled on the corresponding address family.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	[**no**]**router ospf** *process* **Example:** `switch(config)# router ospf test`	Configures OSPF.

Step 3	**segment-routing**
	Example:
	`switch(config-router)# segment-routing`
	`switch(config-sr)#mpls`
	`switch(config-sr-mpls)#`
		Configures the segment routing functionality under OSPF.

Step 4	**interface loopback** *interface_number* **Example:** `switch(config-sr-mpls)# Interface loopback 0`	Specifies the interface where OSPF is enabled.
Step 5	**ip address 1.1.1.1/32** **Example:** `switch(config-sr-mpls)# ip address 1.1.1.1/32`	Specifies the IP address configured on the ospf interface.
Step 6	**ip router ospf 1 area 0** **Example:** `switch(config-sr-mpls)# ip router ospf 1 area 0`	Specifies the OSPF enabled on the interface in area.

Step 7	**segment-routing**
	Example:
	`switch(config-router)#segment-routing`
	`(config-sr)#mpls`
		Configures prefix-sid mapping under SR module.

Step 8	**connected-prefix-sid-map** **Example:**	Configures the prefix SID mapping under the segment routing module.

	Command or Action	Purpose

	`switch(config-sr-mpls)# connected-prefix-sid-map`
	`switch(config-sr-mpls-conn-pfxsid)#`	

Step 9	**address-family ipv4**
	Example:
	`switch(config-sr-mpls-conn-pfxsid)# address-family ipv4`
	`switch(config-sr-mpls-conn-pfxsid-af)#`
		Specifies the IPv4 address family configured on the OSPF interface.

Step 10	**1.1.1.1/32 index 10** **Example:** `switch(config-sr-mpls-conn-af)# 1.1.1.1/32 index 10`	Associates SID 10 with the address 1.1.1.1/32.
Step 11	**exit** **Example:** `switch(config-sr-mpls-conn-af)# exit`	Exits segment routing mode and returns to the configuration terminal mode.

Configuring Prefix Attribute N-flag-clear

OSPF advertises prefix SIDs via Extended Prefix TLV in its opaque LSAs. It carries flags for the prefix and one of them is N flag (Node) indicating that any traffic sent along to the prefix is destined to the router originating the LSA. This flag typically marks host routes of router's loopback.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface loopback3** **Example:** `switch(config)# interface loopback3`	Specifies the interface loopback.
Step 3	**ip ospf prefix-attributes n-flag-clear** **Example:** `switch#(config-if)# ip ospf prefix-attributes n-flag-clear`	Clears the prefix N-flag.

Configuration Examples for Prefix SID for OSPF

This example shows the configuration for prefix SID for OSPF.

```
Router ospf 10
     Segment-routing mpls
Interface loop 0
    Ip address 1.1.1.1/32
    Ip router ospf 10 area 0
Segment-routing
   Mpls
      connected-prefix-sid-m
          address-family ipv4
              1.1.1.1/32 index 10
```

Configuring Segment Routing for Traffic Engineering

About Segment Routing for Traffic Engineering

Segment routing for traffic engineering (SR-TE) takes place through a tunnel between a source and destination pair. Segment routing for traffic engineering uses the concept of source routing, where the source calculates the path and encodes it in the packet header as a segment. A Traffic Engineered (TE) tunnel is a container of TE LSPs instantiated between the tunnel ingress and the tunnel destination. A TE tunnel can instantiate one or more SR-TE LSPs that are associated with the same tunnel.

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

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

SR-TE Policies

Segment routing for traffic engineering (SR-TE) uses a “policy” to steer traffic through the network. A SR-TE policy is a container that includes sets of segments or labels. This list of segments can be provisioned by an operator, a stateful PCE. The head-end imposes the corresponding MPLS label stack on a traffic flow to be carried over the SR-TE policy. Each transit node along the SR-TE policy path uses the incoming top label to select the next-hop, pop or swap the label, and forward the packet to the next node with the remainder of the label stack, until the packet reaches the ultimate destination.

A SR-TE policy is uniquely identified by a tuple (color, end-point). A color is represented as a 32-bit number and an end-point is an IPv4 . Every SR-TE policy has a color value. Every policy between the same node pairs requires a unique color value. Multiple SR-TE policies can be created between the same two endpoints by choosing different colors for the policies.

Cisco Nexus 9000 Series switches support the following two types of SR-TE policies:

- Dynamic SR-TE Policy - When you configure dynamic path preference under the SR-TE policy configuration or an on-demand color configuration, the path computation engine (PCE) calculates the path to the destination address. Dynamic path calculation at PCE results in a list of segments/labels that gets applied to the head-end SR-TE policy, hence the traffic gets routed through the network by hitting the segments that the SR-TE policy holds.

- Explicit SR-TE Policy - An explicit path is a list of labels, each representing a node or link in the explicit path. This feature is enabled through the **explicit-path** command that allows you to create an explicit path and enter a configuration submode for specifying the path.

SR-TE Policy Paths

A SR-TE policy path is a list of segments that specifies the path, called a segment ID (SID) list. Every SR-TE policy consists of one or more candidate paths, which can be either a dynamic or an explicit path. The SR-TE policy instantiates a single path and the selected path is the preferred valid candidate path.

You can also add on-demand color with dynamic path option and explicit policy configuration with an explicit path option for the same color and endpoint. In this case, a single policy is created on the head-end and the path with the highest preference number configured is used for forwarding traffic.

The following two methods are used to compute the SR-TE policy path:.

- Dynamic Path - When you specify the dynamic PCEP option while configuring the path preference under an on-demand color configuration or a policy configuration, the path computation is delegated to a path computation engine(PCE).
- Explicit Path - This path is an explicitly specified SID-list or a set of SID-lists.

Beginning with Cisco NX-OS Release 10.2(2)F, you can lockdown or shutdown an SR-TE policy or perform both; shutdown preference(s) of an SR-TE policy or an on-demand color template; force a specific preference to be active path option for SRTE policy; or force path re-optimization for all or a specific SRTE policy. This feature is supported on Cisco Nexus 9300-EX, 9300-FX, 9300-FX2, 9300-GX, and N9K-C9332D-GX2B platform switches. For more information, see Configuring SR-TE Manual Preference Selection.

For more information about the Cisco Nexus 9000 switches that support various features spanning from release 7.0(3)I7(1) to the current release, refer to Nexus Switch Platform Support Matrix.

Affinity and Disjoint Constraints

Affinity Constraints - You can assign attributes to a link which gets advertised to path computation engine (PCE). SRTE process hosts the affinity-map and interface level configurations. Routing protocol(IGP) will register for interface updates and SRTE will notify IGP with interface updates. IGP tlvs will be passed to BGP to advertise it to external peers. There are three types of affinity constraints:

- exclude-any: specifies that links that have any of the specified affinity colors must not be traversed by the path.
- include-any: specifies that only links that have any of the specified affinity colors must be traversed by the path. Thus, links that do not have any of the specified affinity colors must not be used.
- include-all: specifies that only links that have all of the specified affinity colors must be traversed by the path. Thus, links that do not have all of the specified affinity colors must not be used.

Disjoint Constraints - You can assign disjoint constraints to the SR-TE policies which gets advertised to the PCE. The PCE then provides the disjoint path for the policies that share the same association group ID and the disjoint disjointness type.

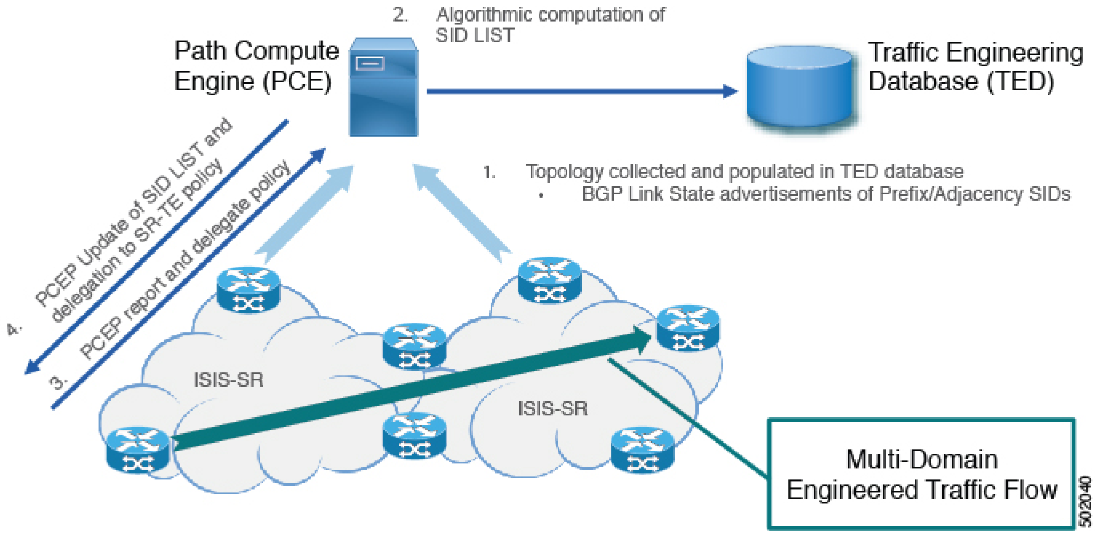
Cisco NX-OS Release 9.3(1) supports the following disjoint path levels :

- Link – The paths transit different links (but may transit same nodes).
- Node disjointness – The paths transit different links but may transit same node.

Segment Routing On Demand Next Hop

On-Demand Next hop (ODN) leverages upon BGP Dynamic SR-TE capabilities and adds the path computation (PCE) ability to find and download the end to end path based on the requirements. ODN triggers an SR-TE auto-tunnel based on the defined BGP policy. As shown in the following figure, an end-to-end path between ToR1 and AC1 can be established from both ends based on IGP Metric. The work-flow for ODN is summarized as follows:

Figure 1: ODN Operation


2. Algorithmic computation of SID LIST
Path Compute Engine (PCE)
Traffic Engineering Database (TED)
1. Topology collected and populated in TED database
• BGP Link State advertisements of Prefix/Adjacency SIDs
4. PCEP Update of SID LIST and delegation to SR-TE policy
3. PCEP report and delegate policy
ISIS-SR
ISIS-SR
Multi-Domain Engineered Traffic Flow
502040

Guidelines and Limitations for SR-TE

SR-TE has the following guidelines and limitations:

- SR-TE ODN for both, IPv4 and IPv6 overlay is supported.
- SR-TE ODN is supported only with IS-IS underlay.
- Forwarding does not support routes with recursive next hops, where the recursive next hop resolves to a route with a binding SID.
- Forwarding does not support mixing paths with binding labels and paths without binding labels for the same route.
- The affinity and disjoint constraints are applicable only to those SR-TE policies that have a dynamic PCEP option.
- XTC supports only two policies with disjointness in the same group.
- When configuring the SR-TE affinity interfaces, the interface range is not supported.

- A preference cannot have both, the dynamic PCEP and the explicit segment lists configured together for the same preference.
- Only one preference can have a dynamic PCEP option per policy.
- For explicit policy, when configuring ECMP paths under same preference, if the first hop (NHLFE) is same for both the ECMP paths, ULIB will only install one path in switching. This occurs because both the ECMP paths create the same SRTE FEC as the NHLFE is same for both.
- In Cisco NX-OS Release 9.3(1), unprotected mode with affinity configuration is not supported by PCE (XTC).
- Beginning with Cisco NX-OS Release 9.3(3), SR-TE ODN, policies, policy paths, and the affinity and disjoint constraints are supported on Cisco Nexus 9364C-GX, Cisco Nexus 9316D-GX, and Cisco Nexus 93600CD-GX switches.
- Beginning with Cisco NX-OS Release 10.2(2)F, few new show commands for SR-TE policy are introduced and the autocomplete feature is provided for some of the existing SR-TE policy commands to improve usability. This feature is supported on Cisco Nexus 9300-EX, 9300-FX, 9300-FX2, 9300-GX, and N9K-C9332D-GX2B platform switches.

Note For more information about the Cisco Nexus 9000 switches that support various features spanning release 7.0(3)I7(1) to the current release, refer to Nexus Switch Platform Support Matrix.

Configuring SR-TE

You can configure segment routing for traffic engineering.

Before you begin

You must ensure that the mpls segment routing feature is enabled.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**segment-routing**	Enters the segment-routing mode
Step 3	**traffic-engineering**	Enters the traffic engineering mode.
Step 4	**encapsulation mpls source ipv4** *tunnel_ip_address*	Configures the source address for the SR-TE Tunnel.
Step 5	**pcc**	Enters the PCC mode.
Step 6	**source-address ipv4** *pcc_source_address*	Configure source address for the PCC
Step 7	**pce-address ipv4 pce_source_address** *precedence* **num**	Configure IP address of the PCE. The lowest numbered PCE will take precedence, and the other(s) be used as a backup.

	Command or Action	Purpose
Step 8	**on-demand color** *color_num*	Enters the on-demand mode to configure the color.
Step 9	**candidate-paths**	Specifies the candidate paths of the policy.
Step 10	**preference** *preference_number*	Specifies the preference of the candidate path.
Step 11	**dynamic**	Specifies the path option.
Step 12	**pcep**	Specifies the path computation that needs to be done from the PCE.

Configuring Affinity Constraints

You can configure the affinity constraints to the SR-TE policy.

Before you begin

You must ensure that the mpls segment routing feature is enabled.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**segment-routing**
	Example:
	`switch(config)# segment-routing`
	`switch(config-sr)#`
		Enables the MPLS segment routing functionality.

Step 3	**traffic-engineering**
	Example:
	`switch(config-sr)# traffic-engineering`
	`switch(config-sr-te)#`
		Enters the traffic engineering mode.

Step 4	**pcc**	Enters the PCC mode.
Step 5	**source-address ipv4** *pcc_source_address*	Configure source address for the PCC
Step 6	**pce-address ipv4 pce_source_address** *precedence* **num**	Configure IP address of the PCE. The lowest numbered PCE takes precedence and the other(s) are used as a backup.
Step 7	**affinity-map** **Example:**	Configures the affinity-map configuration mode.

	Command or Action	Purpose

	`switch(config-sr-te)#affinity-map`
	`switch(config-sr-te-affmap)#`	

Step 8	**color** *name* **bit-position** *position*
	Example:
	`switch(config-sr-te-affmap)# color red bit-position 2`
	`switch(config-sr-te-affmap)#`
		Configures a mapping of the user-defined name to a specific bit position in the affinity bit-map.

Step 9	**interface** *interface-name*
	Example:
	`Enter SRTE interface config mode`
	`switch(config-sr-te-if)#interface eth1/1`
	`switch(config-sr-te-if)#`
		Specifies the name of the interface. This is the affinity mapping name which refers to the specific bit in the affinity bitmap.

Step 10	**affinity**
	Example:
	`switch(config-sr-te-if)# affinity`
	`switch(config-sr-te-if-aff)#`
	`switch(config-sr-te-if-aff)# color red`
	`switch(config-sr-te-if-aff)#`
		Adds the affinity color to the interface.

Step 11	**policy** *name* | **on-demand color** *color_num*
	Example:
	`switch(config-sr-te)# on-demand color 211`
	or
	`switch(config-sr-te-color)# policy test_policy`
		Configures the policy.

Step 12	**color** *color end-point address* **Example:** `switch(config-sr-te-pol)#color 200 endpoint 2.2.2.2`	Configures the color and the end point of the policy. This is required when you are configuring the policy using the "policy name" config mode.

Step 13	**candidate-path**
	Example:
	`switch(config-sr-te-color)# candidate-paths`
	`switch(cfg-cndpath)#`
		Specifies the candidate paths for the policy.

Step 14	**preference** *preference_number*
	Example:
	`switch(cfg-cndpath)# preference 100`
	`switch(cfg-pref)#`
		Specifies the preference of the candidate path.

Step 15	**dynamic** **Example:**	Specifies the path option.

	Command or Action	Purpose

	`switch(cfg-pref)# dynamic`
	`switch(cfg-dyn)#`	

Step 16	**pcep**
	Example:
	`switch(cfg-dyn)# pcep`
	`switch(cfg-dyn)#`
		Specifies that the headend uses PCEP to request the PCE to compute a path from itself to the segment routing's policy's end point.

Step 17	**constraints**
	Example:
	`switch(cfg-dyn)# constraints`
	`switch(cfg-constraints)#`
		Enters the candidate path preference constraint mode.

Step 18	**affinity**
	Example:
	`switch(cfg-constraints)# affinity`
	`switch(cfg-const-aff)#`
		Specifies the affinity constraints of the policy.

Step 19	**exclude-any |include-all | include-any**
	Example:
	`switch(cfg-const-aff)# include-any`
	`switch(cfg-aff-inclany)#`
		Specifies the affinity constraint type. The following affinity types are available:
		• exclude-any - specifies that links that have any of the specified affinity colors must not be traversed by the path.
		• include-any - specifies that only links that have any of the specified affinity colors must be traversed by the path.
		• include-all - specifies that only links that have all of the specified affinity colors must be traversed by the path.

Step 20	**color** *color_name*
	Example:
	`switch(cfg-aff-inclany)# color blue`
	`switch(cfg-aff-inclany)#`
		Specifies the affinity color definition.

Configuring Disjoint Paths

You can configure disjoint path constraints to the SR-TE policy.

Before you begin

You must ensure that the mpls segment routing feature is enabled.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**segment-routing**
	Example:
	`switch(config)# segment-routing`
	`switch(config-sr)#`
		Enables the MPLS segment routing functionality.

Step 3	**traffic-engineering**
	Example:
	`switch(config-sr)# traffic-engineering`
	`switch(config-sr-te)#`
		Enters the traffic engineering mode.

Step 4	**pcc**	Enters the PCC mode.
Step 5	**source-address ipv4** *pcc_source_address*	Configure source address for the PCC
Step 6	**pce-address ipv4 pce_source_address** *precedence* **num**	Configure IP address of the PCE. The lowest numbered PCE takes precedence and the other(s) are used as a backup.

Step 7	**policy** *name* | **on-demand color** *color_num*
	Example:
	`switch(config-sr-te)# on-demand color 211`
	or
	`switch(config-sr-te-color)# policy test_policy`
		Configures the policy.

Step 8	**color** *color end-point address* **Example:** `switch2(config-sr-te-pol)# color 200 endpoint 2.2.2.2`	Configures the color and the end point of the policy. This is required when you are configuring the policy using the “policy name” config mode.

Step 9	**candidate-path**
	Example:
	`switch(config-sr-te-color)# candidate-paths`
	`switch(cfg-cndpath)#`
		Specifies the candidate-paths for the policy

Step 10	**preference** *preference_number*
	Example:
	`switch(cfg-cndpath)# preference 100`
	`switch(cfg-pref)#`
		Specifies the preference of the candidate path.

	Command or Action	Purpose

Step 11	**dynamic**
	Example:
	`switch(cfg-pref)# dynamic`
	`switch(cfg-dyn)#`
		Specifies the path option.

Step 12	**pcep**
	Example:
	`switch(cfg-dyn)# pcep`
	`switch(cfg-dyn)#`
		Specifies that the headend uses PCEP to request the PCE to compute a path from itself to the segment routing's policy's end point.

Step 13	**constraints**
	Example:
	`switch(cfg-dyn)# constraints`
	`switch(cfg-constraints)#`
		Enters the candidate path preference constraint mode.

Step 14	**association-group**
	Example:
	`switch(cfg-constraints)#`
	`association-group`
	`switch(cfg-assoc)#`
		Specifies the association group type.

Step 15	**disjoint**
	Example:
	`switch(cfg-assoc)# disjoint`
	`switch(cfg-disj)#`
		Specifies the path that belongs to the disjointness association group.

Step 16	**type | link | node** **Example:** `switch(config-if)#type link`	Specifies the disjointness group type.
Step 17	**id** *number* **Example:** `switch(config-if)#id 1`	Specifies the identifier of the association-group.

Configuration Examples for SR-TE

The examples in this section show affinity and disjoint configurations.

This example shows the mappings of a user defined name to an administrative group.

```
segment-routing
 traffic-eng
  affinity-map
   color green bit-position 0
   color blue bit-position 2
   color red bit-position 3
```

This example shows the affinity link colors red and green for the adjacency on eth1/1 and affinity link color green for the adjacency on eth1/2.

```
segment-routing
 traffic-eng
  interface eth1/1
   affinity
    color red
    color green
  !
  interface eth1/2
   affinity
    color green
```

This examples shows the affinity constraints for the policy.

```
segment-routing
  traffic-engineering
    affinity-map
      color blue bit-position 0
      color red  bit-position 1
    on-demand color 10
      candidate-paths
        preference 100
          dynamic
            pcep
          constraints
            affinity
              [include-any|include-all|exclude-any]
                color <col_name>
                color <col_name>
    policy new_policy
      color 201 endpoint 2.2.2.0
      candidate-paths
        preference 200
          dynamic
            pcep
          constraints
            affinity
              include-all
                color red
```

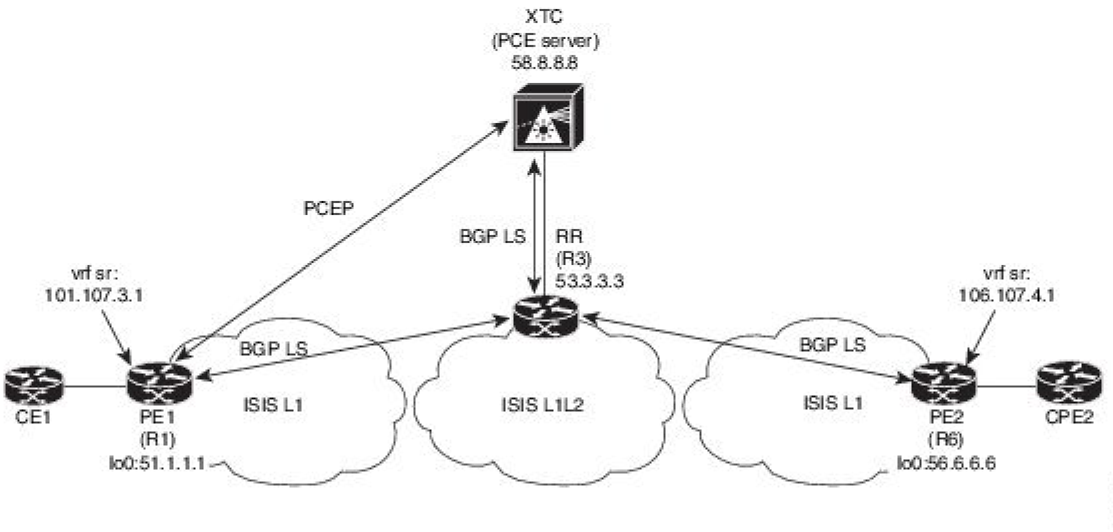
This examples shows the disjoint constraints for the policy.

```
segment-routing
 traffic-eng
  on-demand color 99
   candidate-paths
     preference 100
       dynamic
         pcep
       constraints
         association-group
           disjoint
             type link
             id 1
```

Configuration Example for an SR-TE ODN - Use Case

Perform the following steps to configure ODN for SR-TE. The following figure is used as a reference to explain the configuration steps.

Figure 2: Reference Topology



1. Configure all links with IS-IS point-to-point session from PE1 to PE2. Also, configure the domains as per the above topology.

2. Enable “distribute link-state” for IS-IS session on R1, R3, and R6.

```
router isis 1
  net 31.0000.0000.0000.712a.00
  log-adjacency-changes
  distribute link-state
  address-family ipv4 unicast
    bfd
    segment-routing mpls
    maximum-paths 32
    advertise interface loopback0
```

3. Configure the router R1 (headend) and R6 (tailend) with a VRF interface.

   VRF configuration on R1:

```
interface Ethernet1/49.101
encapsulation dot1q 201
  vrf member sr
  ip address 101.10.1.1/24
  no shutdown

vrf context sr
  rd auto
  address-family ipv4 unicast
    route-target import 101:101
    route-target import 101:101 evpn
    route-target export 101:101
    route-target export 101:101 evpn
router bgp 6500
  vrf sr
    bestpath as-path multipath-relax
    address-family ipv4 unicast
      advertise l2vpn evpn
```

4. Tags VRF prefix with BGP community on R6 (tailend).

```
route-map color1001 permit 10
  set extcommunity color 1001
```

5. Enable BGP on R6 (tailend) and R1 (headend) to advertise and receive VRF SR prefix and match on community set on R6 (tailend).

   R6 < EVPN > R3 < EVPN > R1

   BGP Configuration R6:

```
router bgp 6500
  address-family ipv4 unicast
     allocate-label all
  neighbor 53.3.3.3
    remote-as 6500
    log-neighbor-changes
    update-source loopback0
    address-family l2vpn evpn
      send-community extended
     route-map Color1001 out
      encapsulation mpls
```

   BGP Configuration R1:

```
router bgp 6500
  address-family ipv4 unicast
     allocate-label all
  neighbor 53.3.3.3
    remote-as 6500
    log-neighbor-changes
    update-source loopback0
    address-family l2vpn evpn
      send-community extended
       encapsulation mpls
```

6. Enable BGP configuration on R3 and BGP LS with XTC on R1, R3.abd

   BGP Configuration R3:

```
router bgp 6500
  router-id 2.20.1.2
address-family ipv4 unicast
allocate-label all
address-family l2vpn evpn
retain route-target all
  neighbor 56.6.6.6
    remote-as 6500
    log-neighbor-changes
    update-source loopback0
    address-family l2vpn evpn
      send-community extended
       route-reflector-client
       route-map NH_UNCHANGED out
      encapsulation mpls
  neighbor 51.1.1.1
    remote-as 6500
    log-neighbor-changes
    update-source loopback0
    address-family l2vpn evpn
      send-community extended
      route-reflector-client
      route-map NH_UNCHANGED out
      encapsulation mpls
neighbor 58.8.8.8
    remote-as 6500
    log-neighbor-changes
    update-source loopback0
    address-family link-state

route-map NH_UNCHANGED permit 10
  set ip next-hop unchanged
```

BGP Configuration R1:

```
router bgp 6500
neighbor 58.8.8.8
              remote-as 6500
               log-neighbor-changes
               update-source loopback0
               address-family link-state
```

BGP Configuration R6:

```
outer bgp 6500
   neighbor 58.8.8.8
    remote-as 6500
       log-neighbor-changes
       update-source loopback0
       address-family link-state
```

7. Enable PCE and SR-TE tunnel configurations on R1.

```
segment-routing
  traffic-engineering
    pcc
      source-address ipv4 51.1.1.1
      pce-address ipv4 58.8.8.8
    on-demand color 1001
      metric-type igp
```