Configure Segment Routing over IPv6 (SRv6)

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

- Segment Routing over IPv6 Base, on page 1
- Configuring SRv6 IS-IS, on page 6
- Configuring SRv6 IS-IS TI-LFA, on page 6
- SRv6-Based IPv4 L3VPN, on page 9
- SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode, on page 13

Segment Routing over IPv6 Base

Segment Routing (SR) can be applied on both MPLS and IPv6 data planes. This feature extends Segment Routing support with IPv6 data plane. In an SR-MPLS enabled network, an MPLS label is used as the Segment Identifier (SID) and the source router chooses a path to the destination and encodes the path in the packet header as a stack of labels. However, in a Segment Routing over IPv6 (SRv6) network, an IPv6 address serves as the SID. The source router encodes the path to destination as an ordered list of segments (list of IPv6 addresses) in the IPv6 packet. To encode an ordered list of IPv6 addresses in an IPv6 packet, a new routing header which is an extension header is used. This new header for SRv6 is called Segment Routing Header (SRH). In an SRv6 enabled network, the active segment is indicated by the destination address of the packet, and the next segment is indicated by a pointer in the SRH.

The following list explains the fields in SRH:

- Next header—Identifies the type of header immediately following the SRH.
- Hdr Ext Len (header extension length)—The length of the SRH in 8-octet units, not including the first 8 octets.
- Segments left—Specifies the number of route segments remaining. That means, the number of explicitly listed intermediate nodes still to be visited before reaching the final destination.
- Last Entry—Contains the index (zero based) of the last element of the segment list.
- Flags—Contains 8 bits of flags.
- Tag—Tag a packet as part of a class or group of packets like packets sharing the same set of properties.
- Segment list—128-bit IPv6 addresses representing the nth segment in the segment list. The segment list encoding starts from the last segment of the SR policy (path). That means the first element of the segment...
list (Segment list [0]) contains the last segment of the SR policy, the second element contains the penultimate segment of the SR policy and so on.

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

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

**Table 1: Example of a Segment Routing Header**

<table>
<thead>
<tr>
<th>Next Header</th>
<th>Hdr Ext Len</th>
<th>Routing Type</th>
<th>Segments Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Entry</td>
<td>Flags</td>
<td>Tag</td>
<td></td>
</tr>
<tr>
<td>Segment List[0]</td>
<td>(128-bit IPv6 address)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segment List[(n)]</td>
<td>(128-bit IPv6 address)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optional Type Length Value objects (variable)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

- **Locator**: This is the first part of the SID with most significant bits and represents an address of a specific SRv6 node.

- **Function**: This is the portion of the SID that is local to the owner node and designates a specific SRv6 function (network instruction) that is executed locally on a particular node, specified by the locator bits.

- **Args**: This field is optional and represents optional arguments to the function.

The locator part can be further divided into two parts:

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

- **Node Id**: This field is the node designator in an SRv6 network and is the least significant bit (LSB) portion of a locator subnet.

**Configuring SRv6 Base**

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

The following restrictions and usage guidelines apply while configuring SRv6.
• All routers in the SRv6 domain should have the same SID block (network designator) in their locator.

• The locator length should be 64-bits long.
  • The SID block portion (MSBs) cannot exceed 40 bits. If this value is less than 40 bits, user should use a pattern of zeros as a filler.
  • The Node Id portion (LSBs) cannot exceed 24 bits.

• Only a single locator is supported and that will be the default locator for SID allocation.

**Enabling SRv6 with Locator**
This example shows how to globally enable SRv6 and configure locator.

```
RP/0/0/CPU0:Router (config) # hw-module profile segment-routing srv6
RP/0/0/CPU0:Router (config) # segment-routing sv6
RP/0/0/CPU0:Router (config-srv6) # locators
RP/0/0/CPU0:Router (config-srv6-locators) # locator Loc1
RP/0/0/CPU0:Router (config-srv6-locator) # prefix 2001:db8:0:a2::/64
```

**Note**
For Cisco NCS5500 Series Routers, you must first use the `hw-module profile segment-routing srv6` command to enable SRv6 functionality. Then, reload the line card after enabling this command.

**Optional: Configuring Encapsulation Parameters**
This example shows how to configure encapsulation parameters when configuring SRv6. These optional parameters include:

• Source Address of outer encapsulating IPv6 header: The default source address for encapsulation is one of the loopback addresses,

• Hop Limit of outer encapsulating IPv6 header: The default value for hop-limit is 255.

```
RP/0/0/CPU0:Router (config) # segment-routing sv6
RP/0/0/CPU0:Router (config-srv6) # encapsulation source-address 1::1
RP/0/0/CPU0:Router (config-srv6) # hop-limit 60
```

**Optional: Enabling Syslog Logging for Locator Status Changes**
This example shows how to enable the logging of locator status.

```
RP/0/0/CPU0:Router (config) # segment-routing sv6
RP/0/0/CPU0:Router (config-srv6) # logging locator status
```

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

```
RP/0/0/CPU0:Router # show segment-routing sv6 manager
Parameters:
```
SRv6 Enabled: Yes
Encapsulation:
Source Address:
  Configured: 1::1
  Default: 5::5
Hop-Limit: Default
Summary:
  Number of Locators: 1 (1 operational)
  Number of SIDs: 4 (0 stale)
  Max SIDs: 64000
OOR:
  Thresholds: Green 3200, Warning 1920
  Status: Resource Available (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: Yes
  Microloop-Avoidance: No
End Functions:
  End (PSP)
  End.X (PSP)
  End.DX4
  End.DT4
Transit Functions:
  T
  T.Insert.Red
  T.Encaps.Red
Security rules:
  SEC-1
  SEC-2
  SEC-3
  SEC-4
Counters:
  CNT-1
  CNT-3
Signaled Parameters:
  Max-SL : 4
  Max-End-Pop-SRH : 4
  Max-T-Insert : 4
  Max-T-Encap : 5
  Max-End-D : 5
Max SIDs: 64000
SID Holdtime: 30 mins

Verifying SRv6 Locator

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

```bash
RP/0/0/CP00:Router# show segment-routing srv6 locator myLoc1 detail
Name       ID      Prefix              Status
------------------------ ------------------------ -------
myLoc1*     5       2001:db8:0:a2::/64  Up
(*) : is-default
Interface:
  Name: srv6-myLoc1
  IFH : 0x000000170
  IPv6 address: 2001:db8:0:a2::/64
  Chkpt Obj ID: 0x2fc8
  Created: Apr 25 06:21:57.077 (00:03:37 ago)
```
Verifying SRv6 SIDs

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

```
RP/0/0/CPU0:Router# show segment-routing srv6 locator myLoc1 sid
```

<table>
<thead>
<tr>
<th>SID</th>
<th>Function</th>
<th>Context</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001:db8:0:a2:1::</td>
<td>End (PSP)</td>
<td>'default':1</td>
<td>sidmgr</td>
</tr>
<tr>
<td></td>
<td>InUse Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2001:db8:0:a2:40::</td>
<td>End.DT4</td>
<td>'VRF1'</td>
<td>bgp-100</td>
</tr>
<tr>
<td></td>
<td>InUse Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2001:db8:0:a2:41::</td>
<td>End.X (PSP)</td>
<td>[Hu0/1/0/1, Link-Local]</td>
<td>isis-srv6</td>
</tr>
<tr>
<td></td>
<td>InUse Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

```
RP/0/0/CPU0:Router# show segment-routing srv6 locator myLoc1 sid 2001:db8:0:a2:40:: detail
```

<table>
<thead>
<tr>
<th>SID</th>
<th>Function</th>
<th>Context</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001:db8:0:a2:40::</td>
<td>End.DT4</td>
<td>'VRF1'</td>
<td>bgp-100</td>
</tr>
<tr>
<td></td>
<td>InUse Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SID context: { table-id=0xe0000011 ('VRF1':IPv4/Unicast) }
Locator: myLoc1'
Allocation type: Dynamic
Created: Feb 1 14:04:02.901 (3d00h ago)

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

**show Commands**

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

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

Intermediate System-to-Intermediate System (IS-IS) protocol already supports segment routing with MPLS dataplane (SR-MPLS). This feature enables extensions in IS-IS to support Segment Routing with IPv6 data plane (SRv6). The extensions include advertising the SRv6 capabilities of nodes and node and adjacency segments as SRv6 SIDs.

SRv6 IS-IS performs the following functionalities:

1. Interacts with SID Manager to learn local locator prefixes and announces the locator prefixes in the IGP domain.
2. Learns remote locator prefixes from other ISIS neighbor routers and installs the learned remote locator IPv6 prefix in RIB or FIB.
3. Allocate or learn prefix SID and adjacency SIDs, create local SID entries, and advertise them in the IGP domain.

Restrictions and Usage Guidelines

The following restrictions and usage guidelines apply for SRv6 IS-IS in Cisco IOS XR release 6.6.1.

- Segment routing for traffic engineering (SR-TE) is not supported.
- Segment routing microloop avoidance is not supported.
- An ISIS address-family can support either SR-MPLS or SRv6, but both at the same time is not supported.

Configuring SRv6 IS-IS

To configure SRv6 IS-IS, you should enable SRv6 under the IS-IS IPv6 address-family. The following example shows how to configure SRv6 IS-IS.

```plaintext
RP/0/0/CPU0:Router(config)# router isis srv6
RP/0/0/CPU0:Router(config-isis)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-af)# segment-routing srv6
RP/0/0/CPU0:Router(config-isis-srv6)# locator locator1
RP/0/0/CPU0:Router(config-isis-srv6-loc)# exit
```

Configuring SRv6 IS-IS TI-LFA

This feature introduces support for implementing TI-LFA using IS-IS SRv6.

Restrictions and Usage Guidelines

The following restrictions and usage guidelines apply for this feature in Cisco IOS XR release 6.6.1.

- SRv6 TI-LFA is currently supported only on Cisco NCS5500 Series Routers.
- SRv6 TI-LFA is not supported on both MPLS and SRv6 data planes at the same time for the same IS-IS instance.
• Insertion of up to two SIDs are supported on Cisco NCS5500 Series Routers. If the computed backup path requires more than two SIDs, the backup path will not be installed in the RIB and the path will not be protected.

**Configuring SRv6 IS-IS TI-LFA**

The following example shows how to configure SRv6 IS-IS TI-LFA. You should complete the SRv6 base configuration before performing these steps.

```
RP/0/0/CPU0:Router(config)# router isis srv6
RP/0/0/CPU0:Router(config-isis)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-af)# segment-routing srv6
RP/0/0/CPU0:Router(config-isis-srv6)# locator locator1
RP/0/0/CPU0:Router(config-isis-srv6-loc)# exit
RP/0/0/CPU0:Router(config-isis)# interface loopback 0
RP/0/0/CPU0:Router(config-isis-if)# passive
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# exit
RP/0/0/CPU0:ios(config-isis)# interface bundle-e 1201
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# fast-reroute per-prefix
RP/0/0/CPU0:Router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/0/CPU0:Router(config-isis-if-af)# exit
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# fast-reroute per-prefix
RP/0/0/CPU0:Router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/0/CPU0:Router(config-isis-if-af)# exit
```

**Verification**

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

```
RP/0/RP0/CPU0:Router# show isis ipv6 fast-reroute cafe:0:0:66::/64 detail
Thu Nov 22 16:12:51.983 EST

L1 cafe:0:0:66::/64 [11/115] low priority
  via fe80::2, TenGigE0/0/0/6, SRv6-HUB6, Weight: 0
  Backup path: TI-LFA (link), via fe80::1, Bundle-Ether1201 SRv6-LF1, Weight: 0, Metric: 51
    P node: SRv6-TP8.00 [8::8], SRv6 SID: cafe:0:0:88:1:: End (PSP)
    Backup-src: SRv6-HUB6.00-00, 6::6

P: No, TM: 51, LC: No, NP: No, D: No, SRLG: Yes
```

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

```
RP/0/RP0/CPU0:Router# show route ipv6 cafe:0:0:66::/64 detail
Thu Nov 22 16:14:07.385 EST

Routing entry for cafe:0:0:66::/64
  Known via "isis srv6", distance 115, metric 11, type level-1
  Installed Nov 22 09:24:05.160 for 06:50:02
  Routing Descriptor Blocks
```
This example shows how to verify the SRv6 IS-IS TI-LFA configuration using the `show cef ipv6 ipv6-prefix detail location location` command.

```
RP/0/RP0/CP00:Router# show cef ipv6 cafe:0:0:66::/64 detail location 0/0/cpu0
Thu Nov 22 17:01:58.536 EST
cafe:0:0:66::/64, version 1356, SRv6 Transit, internal 0x1000001 0x2 (ptr 0x8a4a45cc) [1], 0x0 (0x8a46ae20), 0x0 (0x8c8f31b0)
Updated Nov 22 09:24:05.166
local adjacency fe80::2
Prefix Len 64, traffic index 0, precedence n/a, priority 2
gateway array (0x8a2dfaf0) reference count 4, flags 0x500000, source rib (7), 0 backups
  [5 type 3 flags 0x8401 (0x8a395d58) ext 0x0 (0x0)]
LW-LDI[type=3, refc=1, ptr=0x8a46ae20, sh-ldi=0x8a395d58]
gateway array update type-time 1 Nov 22 09:24:05.163
LDI Update time Nov 22 09:24:05.163
LW-LDI-NS Nov 22 09:24:05.163
via fe80::2/128, TenGigE0/0/0/6, 8 dependencies, weight 0, class 0, protected [flags 0x400]
  path-idx 0 bkp-idx 1 NHID 0x2000a [0x8a2c2fd0 0x00]
nexthop fe80::2/128
via fe80::1/128, Bundle-Ether1201, 8 dependencies, weight 0, class 0, backup (TI-LFA) [flags 0xb00]
  path-idx 1 NHID 0x2000d [0x8c2670b0 0x00]
nexthop fe80::1/128, Repair Node(s): 8::8
local adjacency
SRv6 T.Insert.Red SID-list { cafe:0:0:88:1:: }
MPLS eid:0x1380B00000001
```

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

```
RP/0/RP0/CP00:Router# show cef ipv6 fast-reroute-db
```
SRv6-Based IPv4 L3VPN

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

For this feature, BGP allocates an SRv6 SID from the locator space, configured under SRv6-base and VPNv4 address family. For more information on this, refer Segment Routing over IPv6 Base, on page 1. The BGP SID can be allocated in the following ways:

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

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

Restrictions and Usage Guidelines

- MPLS based L3VPN inter-operability is not supported on a router that is configured for SRv6-based L3VPN.
- Only IPv4 L3VPN is supported, and IPv6 L3VPN is not supported.
- Equal-Cost Multi-path (ECMP) and Unequal Cost Multipath (UCMP) are supported.
- BGP, OSPF, Static are supported as PE-CE protocol.
Configuring SRv6 based IPv4 L3VPN

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

/*Configure BGP and specify locator name*/
RP/0/CPU0:Router(config)# router bgp 100
RP/0/CPU0:Router(config-bgp)# bgp router-id 10.6.6.6
RP/0/CPU0:Router(config-bgp)# address-family vpnv4 unicast
RP/0/CPU0:Router(config-bgp-af)# segment-routing srv6
RP/0/CPU0:Router(config-bgp-af-srv6)# locator my-locator
RP/0/CPU0:Router(config-bgp-af-srv6)# exit

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

/*Configure a VRF with per-ce label allocation mode*/
RP/0/CPU0:Router(config-bgp-af)# vrf vrf2
RP/0/CPU0:Router(config-bgp-vrf)# rd 106:2
RP/0/CPU0:Router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/CPU0:Router(config-bgp-vrf-af)# segment-routing srv6
RP/0/CPU0:Router(config-bgp-vrf-af-srv6)# alloc mode per-ce
RP/0/CPU0:Router(config-bgp-vrf-af-srv6)# exit
RP/0/CPU0:Router(config-bgp-vrf-af-srv6)# exit
RP/0/CPU0:Router(config-bgp-vrf)# neighbor 10.1.2.2
RP/0/CPU0:Router(config-bgp-vrf-nbr)# remote-as 100
RP/0/CPU0:Router(config-bgp-vrf-nbr)# address-family ipv4 unicast

Verification

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

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

RP/0/CPU0:Router# show segment-routing srv6 sid
*** Locator: 'my_locator' ***

<table>
<thead>
<tr>
<th>SID</th>
<th>State</th>
<th>Function</th>
<th>Context</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>cafe:0:0:66:1::</td>
<td>InUse</td>
<td>End (PSP)</td>
<td>'my_locator':1</td>
<td>sidmgr</td>
</tr>
<tr>
<td>cafe:0:0:66:40::</td>
<td>InUse</td>
<td>End.X (PSP)</td>
<td>[Te0/0/0/2, Link-Local]</td>
<td>isis-srv6</td>
</tr>
<tr>
<td>cafe:0:0:66:41::</td>
<td>InUse</td>
<td>End.X (PSP)</td>
<td>[BE6801, Link-Local]</td>
<td>isis-srv6</td>
</tr>
<tr>
<td>cafe:0:0:66:42::</td>
<td>InUse</td>
<td>End.X (PSP)</td>
<td>[BE5601, Link-Local]</td>
<td>isis-srv6</td>
</tr>
<tr>
<td>cafe:0:0:66:43::</td>
<td>InUse</td>
<td>End.X (PSP)</td>
<td>[BE5602, Link-Local]</td>
<td>isis-srv6</td>
</tr>
</tbody>
</table>
The following example shows how to verify the SRv6 based L3VPN configuration using the `show segment-routing srv6 SID-prefix detail` command.

```
RP/0/RP0/CPU0:Router#sh segment-routing srv6 sid cafe:0:0:66:43:: detail
Sun Dec 9 16:52:54.015 EST
*** Locator: 'my_locator' ***
SID Function Context Owner
-------------------------- ----------- ------------------------------ ------------------
----- --
cafe:0:0:66:44:: End.DT4 'VRF1' bgp-100
```

```
RP/0/RP0/CPU0:SRv6-HUB6#sh segment-routing srv6 sid cafe:0:0:66:47:: detail
Sun Dec 9 16:54:26.073 EST
*** Locator: 'my_locator' ***
SID Function Context Owner
-------------------------- ----------- ------------------------------ ------------------
----- --
cafe:0:0:66:45:: End.DX4 'VRF2':4 bgp-100
```

The following example shows how to verify the SRv6 based L3VPN configuration using the `show bgp vpnv4 un rd` command on Egress PE.

```
RP/0/RP0/CPU0:SRv6-Hub6#sh bgp vpnv4 un rd 106:1 10.15.0.0/30
Wed Nov 21 16:08:44.765 EST
BGP routing table entry for 10.15.0.0/30, Route Distinguisher: 106:1
Versions:
    Process       dRIB/RIB  SendTblVer
    Speaker 2282449 2282449
SRv6-VPN SID: cafe:0:0:66:44::/128
Last Modified: Nov 21 15:50:34.235 for 00:18:10
Paths: (2 available, best #1)
    Advertised to peers (in unique update groups):
        2::2
    Path #1: Received by speaker 0
    Advertised to peers (in unique update groups):
        2::2
200
    10.1.2.2 from 10.1.2.2 (10.7.0.1)
        Origin IGP, localpref 200, valid, internal, best, group-best, import-candidate
        Received Path ID 0, Local Path ID 1, version 2276228
        Extended community: RT:201:1
```
The following example shows how to verify the SRv6 based L3VPN configuration using the `show bgp vpng un rd route-distinguisher prefix` command on Ingress PE.

```
RP/0/RP0/CPU0:SRv6-LF1#sh bgp vpnv4 un rd 106:1 10.15.0.0/30
Wed Nov 21 16:11:45.538 EST
BGP routing table entry for 10.15.0.0/30, Route Distinguisher: 106:1
Versions:
  Process bRIB/RIB SendTblVer
  Speaker 2286222 2286222
Last Modified: Nov 21 15:47:26.288 for 00:24:19
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  200, (received & used)
  6::6 (metric 24) from 2::2 (6.6.6.6)
  Received Label 3
  Origin IGP, localpref 200, valid, internal, best, group-best, import-candidate, not-in-vrf
  Received Path ID 1, Local Path ID 1, version 2286222
  Extended community: RT:201:1
  Originator: 6.6.6.6, Cluster list: 2.2.2.2
SRv6-VPN-SID: T1-cafe:0:0:66:44:: [total 1]
```

The following example shows how to verify the SRv6 based L3VPN configuration using the `show route vrf vrf-name prefix detail` command.

```
RP/0/RP0/CPU0:Router#sh route vrf VRF1 10.15.0.0/30 detail
Wed Nov 21 16:35:17.775 EST
Routing entry for 10.15.0.0/30
Known via "bgp 100", distance 200, metric 0
Tag 200, type internal
Installed Nov 21 16:35:14.107 for 00:00:03
Routing Descriptor Blocks
  6::6, from 2::2
    Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
    Route metric is 0
    Label: None
    Tunnel ID: None
    Binding Label: None
    Extended communities count: 0
    Source RD attributes: 0x0000:106:1
    NHID:0x0(Ref:0)
    SRv6 Transit Type: T.Encaps.Red
    SRv6 SID-list { cafe:0:0:66:44:: }
    MPLS eid:0x138060000001
    Route version is 0xd (13)
    No local label
    IP Precedence: Not Set
    QoS Group ID: Not Set
    Flow-tag: Not Set
    Fwd-class: Not Set
    Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type RIB_SVD_TYPE_REMOTE
    Download Priority 3, Download Version 3038384
    No advertising protos.
```
The following example shows how to verify the SRv6 based L3VPN configuration for per-ce allocation mode using the `show bgp vrf vrf nexthop-set` command.

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

The following example shows how to verify the SRv6 based L3VPN configuration using the `show cef vrf vrf-name prefix detail location line-card` command.

```
RP/0/RP0/CPU0:Router#sh cef vrf VRF1 10.15.0.0/30 detail location 0/0/cpu0
Wed Nov 21 16:37:06.894 EST
151.1.0.0/30, version 3038384, SRv6 Transit, internal 0x50000001 0x0 (ptr 0x9ae6474c) [1], 0x0 (0x0), 0x0 (0x8c11b238)
Updated Nov 21 16:35:14.109
Prefix Len 30, traffic index 0, precedence n/a, priority 3
gateway array (0x8cd85190) reference count 1014, flags 0x2010, source rib (7), 0 backups
[1 type 3 flags 0x40441 (0x8a529798) ext 0x0 (0x0)]
LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
gateway array update type-time 1 Nov 21 14:47:26.816
LDI Update time Nov 21 14:52:53.073
Level 1 - Load distribution: 0 [0] via cafe:0:0:66::/128, recursive
via cafe:0:0:66::/128, 7 dependencies, recursive [flags 0x6000]
path-idx 0 NHID 0x0 [0x8acb53cc 0x0]
next hop VRF - 'default', table - 0xe0800000
next hop cafe:0:0:66::128 via cafe:0:0:66::64
SRv6 T.Encaps.Red SID-list {cafe:0:0:66:44::}
Load distribution: 0 (refcount 1)
Hash OK Interface Address
0 Y Bundle-Ether1201 fe80::2
```

**SRv6 Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode**

The Segment Routing IPv6 (SRv6) Services: L3VPN VPNv4 Active-Standby Redundancy using Port-Active Mode feature provides all-active per-port load balancing for multihoming. The forwarding of traffic is determined based on a specific interface rather than per-flow across multiple Provider Edge routers. This feature enables efficient load-balancing and provides faster convergence. In an active-standby scenario, the active PE router is detected using designated forwarder (DF) election by modulo calculation and the interface of the standby PE router brought down. For Modulo calculation, byte 10 of the Ethernet Segment Identifier (ESI) is used.
Restrictions

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

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

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

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

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

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

Configuration Example

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

/* Configure load balancing. */
Router# configure
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether10
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 11.11.11.11.11.11.11.14
Router(config-evpn-ac-es)# load-balancing-mode port-active
Router(config-evpn-ac-es)# commit

/* Configure address family session in BGP. */
Router# configure
Router(config)# router bgp 100
Router(config-router)# bgp router-id 192.168.0.2```
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp)# neighbor 192.168.0.3
Router(config-bgp-nbr)# remote-as 200
Router(config-bgp-nbr)# update-source Loopback 0
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr)# commit

### Running Configuration

```
interface Bundle-Ether14
 ipv4 address 14.0.0.2 255.255.255.0
 ipv6 address 14::2/64
 lacp period short
 mac-address 1.2.3
 bundle wait-while 0
!
interface GigabitEthernet0/2/0/5
 bundle id 14 mode active
!
evpn
 interface Bundle-Ether14
   ethernet-segment
     identifier type 0 11.1111.1111.1111.1114
     load-balancing-mode port-active
!
!
routing bgp 100
 bgp router-id 192.168.0.2
 address-family l2vpn evpn
!
neighbor 192.168.0.3
 remote-as 100
 update-source Loopback0
 address-family l2vpn evpn
!
!
```

### Verification

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

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

 ES to BGP Gates : Ready
 ES to L2FIB Gates : Ready
 Main port :
   Interface name : Bundle-Ether14
   Interface MAC : 0001.0002.0003
   IfHandle : 0x0000041d0
   State : Up
   Redundancy : Not Defined
 ESI type : 0
   Value : 11.1111.1111.1111.1114
 ES Import RT : 1111.1111.1111 (from ESI)
 Source MAC : 0000.0000.0000 (N/A)
```
Topology:
  Operational: MH
  Configured: Port-Active
Service Carving: Auto-selection
  Multicast: Disabled
Peering Details:
  192.168.0.2 [MOD:P:00]
  192.168.0.3 [MOD:P:00]
Service Carving Results:
  Forwarders: 0
  Permanent: 0
  Elected: 0
  Not Elected: 0
MAC Flushing mode: STP-TCN
Peering timer: 3 sec [not running]
Recovery timer: 30 sec [not running]
Carving timer: 0 sec [not running]
Local SHG label: None
Remote SHG labels: 0

/* Verify bundle Ethernet configuration on active DF router */
Router# show bundle bundle-ether 14
Bundle-Ether14
  Status: Up
  Local links <active/standby/configured>: 1 / 0 / 1
  Local bandwidth <effective/available>: 1000000 (1000000) kbps
  MAC address (source): 0001.0002.0003 (Configured)
  Inter-chassis link: No
  Minimum active links / bandwidth: 1 / 1 kbps
  Maximum active links: 64
  Wait while timer: Off
  Load balancing:
    Link order signaling: Not configured
    Hash type: Default
    Locality threshold: None
  LACP:
    Flap suppression timer: Off
    Cisco extensions: Disabled
    Non-revertive: Disabled
  mLACP: Not configured
  IPv4 BFD: Not configured
  IPv6 BFD: Not configured

<table>
<thead>
<tr>
<th>Port</th>
<th>Device</th>
<th>State</th>
<th>Port ID</th>
<th>B/W, kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gi0/2/0/5</td>
<td>Local</td>
<td>Active</td>
<td>0x8000, 0x0003</td>
<td>1000000</td>
</tr>
</tbody>
</table>

/* Verify ethernet-segment details on standby DF router */
Router# show evpn ethernet-segment interface bundle-ether 10 detail

Router# show evpn ethernet-segment interface Bundle-Ether24 detail
Ethernet Segment Id      Interface         Nexthops
------------------------ ----------------- -----------------
0011.1111.1111.1111.1114 BE24         192.168.0.2
                                    192.168.0.3
ES to BGP Gates    : Ready
ES to L2FIB Gates  : Ready
Main port          :
  Interface name: Bundle-Ether24
Interface MAC : 0001.0002.0003
IfHandle : 0x000041b0
State : Standby
Redundancy : Not Defined
ESI type : 0
Value : 11.1111.1111.1111.114
ES Import RT : 1111.1111.1111 (from ESI)
Source MAC : 0000.0000.0000 (N/A)
Topology :
   Operational : MH
   Configured : Port-Active
Service Carving : Auto-selection
Multicast : Disabled
Peering Details :
   192.168.0.2 [MOD:P:00]
   192.168.0.3 [MOD:P:00]
Service Carving Results:
   Forwarders : 0
   Permanent : 0
   Elected : 0
   Not Elected : 0
MAC Flushing mode : STP-TCN
Recovery timer : 3 sec [not running]
Carving timer : 0 sec [not running]
Local SHG label : None
Remote SHG labels : 0
/* Verify bundle configuration on standby DF router */
Router# show bundle bundle-ether 24

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

<table>
<thead>
<tr>
<th>Port</th>
<th>Device</th>
<th>State</th>
<th>Port ID</th>
<th>B/W, kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G10/0/0/4</td>
<td>Local</td>
<td>Standby</td>
<td>0x8000, 0x0002</td>
<td>100000</td>
</tr>
</tbody>
</table>
   Link is in standby due to bundle out of service state
Verification