SRv6-Based Layer 2 and Integrated VPN Services

IPv4 L3VPN active-standby redundancy using port-active mode

An IPv4 L3VPN active-standby redundancy using port-active mode is a Segment Routing IPv6 (SRv6) service that

- provides all-active per-port load balancing for multihoming
- determines traffic forwarding based on specific interfaces rather than per-flow across multiple Provider Edge (PE) routers, and
- enables faster convergence by using designated forwarder (DF) election through modulo calculation, where byte 10 of the Ethernet Segment Identifier (ESI) is used to detect the active PE router and bring down the standby PE router's interface

Enhanced multihoming efficiency with active-standby redundancy using port-active mode

This feature ensures high availability and efficient traffic management in multihomed IPv4 L3VPN networks. By leveraging per-port active-standby redundancy, it provides robust load balancing and rapid failover, minimizing service disruption. The port-active mode simplifies traffic forwarding decisions based on specific interfaces and accelerates convergence using DF election, enhancing overall network resilience and operational efficiency.

Benefits of active-standby redundancy using port-active mode

- Load balancing performed per interface (port), simplifying forwarding decisions.
- DF election based on modulo calculation of the ESI to identify the active PE.
- Standby PE interfaces disabled to avoid forwarding loops and conflicts.
- Rapid failover and convergence by re-electing the DF upon link or port failure.

- Enhanced network efficiency, reliability, and resilience in multihomed IPv4 L3VPN deployments.

Restrictions of active-standby redundancy using port-active mode

- This feature can only be configured on bundle interfaces.
- When an EVPN Ethernet segment is configured with port-active load-balancing mode, you must not configure ACs of that bundle on bridge domains that have an EVPN instance (EVI) configured.
- EVPN Layer 2 bridging service is incompatible with port-active load-balancing mode and therefore cannot be used together.

How SRv6 services for L3VPN active-standby redundancy using port-active mode work

Summary

The SRv6 Services for L3VPN Active-Standby Redundancy using Port-Active Mode process ensures high availability and efficient traffic management. PE routers exchange EVPN ES routes via BGP, then individually determine an ordered list of PEs. A modulo calculation designates one PE as the active forwarder (DF), while others disable their bundles. In the event of a failure, the active DF withdraws its route, prompting a re-election and quick resumption of forwarding duties by a newly elected DF.

Workflow

These stages describe how SRv6 services for IPv4 L3VPN active-standby redundancy using port-active mode works:

1. PE routers exchange EVPN ES routes. All PE routers connected to the multihomed ES exchange EVPN ES routes across BGP.
2. Each PE router creates an ordered list and assigns an ordinal.
   - Each PE router creates an ordered list of all PE IP addresses connected to the ES.
   - Each PE router assigns itself an ordinal based on its position in this list.
3. A modulo calculation determines the Designated Forwarder (DF).
   - Using the ordinals, a modulo calculation determines which PE becomes the Designated Forwarder (DF) for the ES.
   - For the modulo calculation, byte 10 of the Ethernet Segment Identifier (ESI) is used.
4. The elected DF forwards traffic and non-DF PEs disable bundles.
   - The PE elected as DF actively forwards traffic for the ES.
   - All other PEs (non-DF) disable their respective bundles for that ES.
5. The active DF PE withdraws its ES route upon failure.

   Upon a link or port failure, the active DF PE withdraws its ES route.

6. A new DF election is triggered, and a new PE resumes forwarding.
   - This withdrawal triggers a new DF election among all PEs servicing the ES.
   - A new PE is elected as DF and resumes forwarding duties.

Configure SRv6 services L3VPN active-standby redundancy using port-active mode

Configure SRv6 services to provide L3VPN active-standby redundancy using port-active mode under an Ethernet Segment (ES).

Procedure

Step 1 Configure Ethernet link bundle interface.

Example:

```
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
```

Step 2 Configure the physical interface to join the bundle.

Example:

```
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 11.11.11.11.11.11.11.11.14
Router(config-evpn-ac-es)# load-balancing-mode port-active
Router(config-evpn-ac-es)# commit
```

Step 3 Configure the BGP address family session for EVPN.

Example:

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

Step 4 Running configuration active-standby redundancy using port-active mode.

Example:

```
interface Bundle-Ether14
 ipv4 address 14.0.0.2 255.255.255.0
 ipv6 address 14::2/64
 lacp period short
 mac-address 1.2.3
 bundle wait-while 0
!
interface GigabitEthernet0/2/0/5
 bundle id 14 mode active
!
evpn
 interface Bundle-Ether14
  ethernet-segment
   identifier type 0 11.11.11.11.11.11.11.11.14
   load-balancing-mode port-active
  !
 !
!
router bgp 100
 bgp router-id 192.168.0.2
 address-family l2vpn evpn
 !
 neighbor 192.168.0.3
  remote-as 100
  update-source Loopback0
  address-family l2vpn evpn
  !
 !
!
```

Step 5 Verify the SRv6 services L3VPN active-standby redundancy using port-active mode configuration.

a) Verify ethernet-segment details on active DF router.

Example:

```
Router# show evpn ethernet-segment interface Bundle-Ether14 detail
Ethernet Segment Id      Interface                         Nexthops
------------------------ --------------------------------- --------------------
0011.1111.1111.1111.1114 BE14                              192.168.0.2
                                                           192.168.0.3

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

```
    192.168.0.3 [MOD:P:00]

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

b) Verify bundle Ethernet configuration on active DF router.

Example:

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

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

c) Verify ethernet-segment details on standby DF router.

Example:

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

Router# show evpn ethernet-segment interface Bundle-Ether24 detail
Ethernet Segment Id      Interface                         Nexthops
------------------------ --------------------------------- --------------------
0011.1111.1111.1111.1114 BE24                              192.168.0.2
                                                           192.168.0.3
  ES to BGP Gates   : Ready
  ES to L2FIB Gates : Ready
  Main port         :
     Interface name : Bundle-Ether24
     Interface MAC  : 0001.0002.0003
     IfHandle       : 0x000041b0
     State          : Standby
```

```
    Redundancy     : Not Defined
  ESI type         : 0
    Value          : 11.1111.1111.1111.1114
  ES Import RT     : 1111.1111.1111 (from ESI)
  Source MAC       : 0000.0000.0000 (N/A)
  Topology         :
    Operational    : MH
    Configured     : Port-Active
  Service Carving  : Auto-selection
    Multicast      : Disabled
  Peering Details  :
    192.168.0.2 [MOD:P:00]
    192.168.0.3 [MOD:P:00]

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

d) Verify bundle configuration on standby DF router.

Example:

```
Router# show bundle bundle-ether 24

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

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

IPv4 L3VPN active-active redundancy using port-active mode

An IPv4 L3VPN active-active redundancy using port-active mode is an SRv6 service that

- provides active-active connectivity to a CE device in a Layer 3 VPN (L3VPN) deployment
- allows the CE device, whether Layer 2 or Layer 3, to connect to redundant PE routers over a single Link Aggregation Control Protocol (LACP) link aggregation group (LAG) port, and
- augments Layer 3 local route learning with remote route-synchronization programming to ensure complete route awareness across redundant PEs despite bundle hashing distributing ARP or IPv6 Network Discovery packets unevenly

Key concepts

- Bundle hashing: A method that distributes traffic across multiple links in a LAG based on packet header fields, which can cause packets like ARP or IPv6 ND to be sent to different redundant routers.
- Remote route-synchronization programming: A mechanism that synchronizes Layer 3 routing information between redundant PEs to provide full route awareness

Route synchronization between service PEs

Route synchronization between service PE devices is essential to minimize service interruptions for both unicast and multicast traffic following a failure on a redundant service PE. The synchronization process uses specific EVPN route types for Layer 3 route synchronization:

- EVPN route-type 2: Synchronizes ARP tables.
- EVPN route-type 7/8: Synchronizes IGMP JOIN and LEAVE messages.

In scenarios where a Layer 3 CE router connects to redundant PEs, it may establish an IGP adjacency on the bundle port. This adjacency forms with only one of the redundant PEs, meaning that IGP customer routes are present only on that PE. To synchronize Layer-3 customer subnet routes (IP prefixes) across redundant PEs, EVPN route-type 5 is used. This route type carries the ESI, Ethernet Tag (ETAG), and the gateway address (prefix next-hop address) to ensure consistent routing information.

This synchronization mechanism ensures seamless failover and continuity of Layer 3 services in redundant PE environments

Consistent route synchronization across VRFs and Ethernet segments

Consistent route synchronization across Virtual Routing and Forwarding instances (VRFs) and Ethernet segments ensures optimized redundancy and efficient route management in the network. This synchronization maintains uniform routing information, which enhances network stability and simplifies the management of redundant paths and failover mechanisms.

- Synchronization across VRFs prevents routing inconsistencies between isolated routing domains.
- Ethernet segment synchronization supports redundancy by coordinating route updates among connected devices.
- The approach optimizes route management by reducing conflicts and improving convergence times.

- Gratuitous ARP (GARP) and IPv6 Network Advertisement (NA) replay are not necessary for CE devices connected to redundant PE devices through a single Link LAG port.
- This configuration example enables Layer 3 route synchronization for routes learned on Ethernet segment subinterfaces:

```
evpn
 route-sync vrf default
!
vrf RED
 evi route-sync 10
!
vrf BLUE
 evi route-sync 20
!
```

Note EVPN does not support untagged interfaces.

SRv6 L3 EVPN services

An SRv6 L3 EVPN services is a network service that

- uses EVPN Route Type 5 (RT5) to advertise EVPN routes with IP prefixes
- provides end-to-end Layer 3 connectivity, and
- supports carrying L3VPN routes within the L2VPN EVPN RT5 address family instead of VPNv4 or VPNv6 unicast address families across an SRv6 core (EVPN over SRv6 underlay).

Table 1: Feature History Table

Feature Name	Release	Description
Support for SRv6 Services: L3 EVPN	Release 7.9.1	This feature adds support for carrying L3VPN routes in L2VPN EVPN RT5 address-family instead of VPNv4 unicast and/or VPNv6 unicast address-family across SRv6 core (EVPN over SRv6 underlay). This allows for efficient and scalable delivery of VPN services using SRv6 technology.

Interworking and BGP session for EVPN RT5 over SRv6 and MPLS cores

Interworking between EVPN RT5 over SRv6 core and EVPN RT5 over MPLS core is supported through the L3 EVPN/SRv6 and L3 EVPN/MPLS interworking gateway.

Similarly, interworking between EVPN RT5 over SRv6 core and L3VPN over MPLS core is enabled through the L3 EVPN/SRv6 and L3VPN/MPLS interworking gateway.

Limitations of SRv6 L3 EVPN services

This topic provides supported configurations for interworking EVPN RT5 with SRv6 and MPLS core.

Supported interworking

- Interworking between EVPN RT5 over an SRv6 core and EVPN RT5 over an MPLS core is supported.
- Interworking between EVPN RT5 over an SRv6 core and L3VPN over an MPLS core is supported.

BGP session configuration for route reflectors

- BGP does not support configuring both VPNv4/v6 address families and EVPN RT5 address families on the same BGP session simultaneously.

  For the route reflector (RR) to receive both Type-5 EVPN route and VPNv4/v6 address family, we recommend that you configure two pairs of loopback interfaces and configure two BGP loopback sessions between the RR and the PE: one session for VPNv4/v6 address family and one session for EVPN address family.

VRF route advertisement

- BGP advertises all VRF routes using either the VPNv4/v6 or EVPN address family.

  We recommend that you mark the VRF route via export route-policy and use neighbor out policy to either drop or pass the route for an address family to achieve the same net effect.

Supported behaviors for EVPN RT5 over SRv6

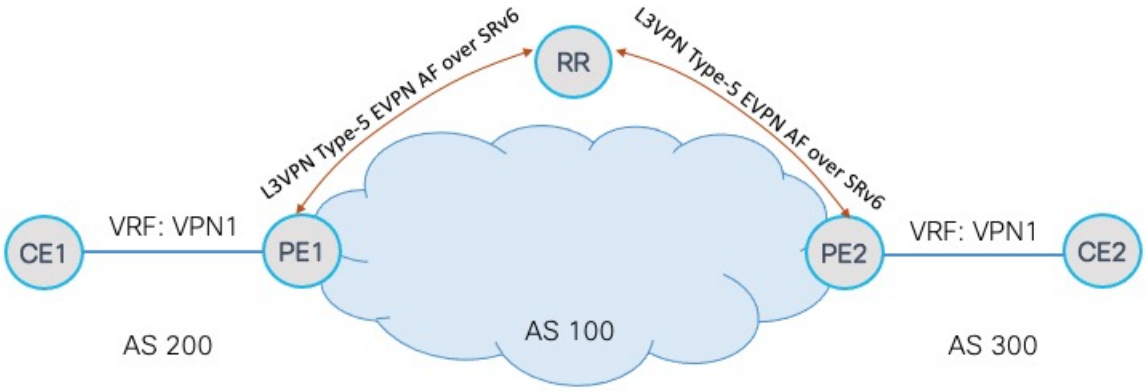
- IPv4, IPv6, and IPv4/IPv6 (dual stack) L3 EVPN over SRv6
- uDT4
- uDT6
- uDT46
- Automated Steering to Flex-Algo (BGP per-VRF locator Flex-Algo (per-prefix))
- Automated Steering to SRv6 Policy (ODN/AS)

Configure SRv6-based L3 EVPN

Enable and configure SRv6-based Layer 3 EVPN to support dual-stack IPv4/IPv6 environments.

-

Figure 1: Configuration Example: Dual Stack L3 EVPN over SRv6



Procedure

Step 1 Configure the VRF with dual-stack IPv4 and IPv6 address families.

Example:

```
Router(config)# vrf VPN1
Router(config-vrf)# address-family ipv4 unicast
Router(config-vrf-af)# import route-target
Router(config-vrf-import-rt)# 1:1
Router(config-vrf-import-rt)# exit
Router(config-vrf-af)# export route-target
Router(config-vrf-export-rt)# 1:1
Router(config-vrf-export-rt)# exit
Router(config-vrf)# address-family ipv6 unicast
Router(config-vrf-af)# import route-target
Router(config-vrf-import-rt)# 1:1
Router(config-vrf-import-rt)# exit
Router(config-vrf-af)# export route-target
Router(config-vrf-export-rt)# 1:1
Router(config-vrf-export-rt)# exit
Router(config-vrf-af)#
```

Step 2 Configure the SRv6 locator for an individual VRF, with per-VRF label allocation mode.

Example:

```
Router(config)# router bgp 100
Router(config-bgp)# address-family vpnv4 unicast
Router(config-bgp-af)# additional-paths receive
Router(config-bgp-af)# additional-paths send
Router(config-bgp-af)# additional-paths selection route-policy add-path
Router(config-bgp-af)# exit
Router(config-bgp)# address-family vpnv6 unicast
Router(config-bgp-af)# additional-paths receive
Router(config-bgp-af)# additional-paths send
Router(config-bgp-af)# additional-paths selection route-policy add-path
Router(config-bgp-af)# exit
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# additional-paths receive
Router(config-bgp-af)# additional-paths send
Router(config-bgp-af)# additional-paths selection route-policy add-path
```

```
Router(config-bgp-af)# exit

Router(config-bgp)# neighbor 1111::1
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# advertise vpnv4 unicast
Router(config-bgp-nbr-af)# advertise vpnv6 unicast
Router(config-bgp-nbr-af)# exit
Router(config-bgp-nbr)# exit

Router(config-bgp)# vrf VPN1
Router(config-bgp-vrf)# rd 100:1
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# segment-routing srv6
Router(config-bgp-vrf-af-srv6)# locator LOC1
Router(config-bgp-vrf-af-srv6)# alloc mode per-vrf
Router(config-bgp-vrf-af-srv6)# exit
Router(config-bgp-vrf-af)# exit
Router(config-bgp-vrf)# address-family ipv6 unicast
Router(config-bgp-vrf-af)# segment-routing srv6
Router(config-bgp-vrf-af-srv6)# locator LOC1
Router(config-bgp-vrf-af-srv6)# alloc mode per-vrf -46
Router(config-bgp-vrf-af-srv6)# exit
Router(config-bgp-vrf-af)# exit

Router(config-bgp-vrf)# neighbor 1.1.1.1
Router(config-bgp-vrf-nbr)# remote-as 200
Router(config-bgp-vrf-nbr)# address-family ipv4 unicast
Router(config-bgp-vrf-nbr-af)# exit
Router(config-bgp-vrf-nbr)# exit
Router(config-bgp-vrf)# neighbor 3333::3
Router(config-bgp-vrf-nbr)# remote-as 200
Router(config-bgp-vrf-nbr)# address-family ipv6 unicast
```

Step 3 Running configuration of SRv6-based L3 EVPN

Example:

```
vrf VPN1
 address-family ipv4 unicast
  import route-target
   1:1
  !
  export route-target
   1:1
  !
 !
 address-family ipv6 unicast
  import route-target
   1:1
  !
  export route-target
   1:1
  !
 !
!
router bgp 100
 address-family vpnv4 unicast
  additional-paths receive
  additional-paths send
  additional-paths selection route-policy add-path
 !
 address-family vpnv6 unicast
```

```
 additional-paths receive
 additional-paths send
 additional-paths selection route-policy add-path
!
address-family l2vpn evpn
 additional-paths receive
 additional-paths send
 additional-paths selection route-policy add-path
!
neighbor 1111::1
 remote-as 100
 address-family l2vpn evpn
  advertise vpnv4 unicast
  advertise vpnv6 unicast
 !
!
vrf VPN1
 rd 100:1
 address-family ipv4 unicast
  segment-routing srv6
   locator LOC1
   alloc mode per-vrf
  !
 !
 address-family ipv6 unicast
  segment-routing srv6
   locator LOC1
   alloc mode per-vrf -46
  !
 !
 neighbor 1.1.1.1
  remote-as 200
  address-family ipv4 unicast
  !
 !
 neighbor 3333::3
  remote-as 200
  address-family ipv6 unicast
  !
 !
!
```

SRv6 service support

Segment Routing over IPv6 (SRv6) enables advanced network programmability and service chaining by steering packets through a network based on instructions encoded as IPv6 addresses, known as Segment IDs (SIDs). Modern SRv6 deployments leverage SID lists and SID databases (such as W-LIB) to support flexible and scalable network services, including both Layer 2 and Layer 3 offerings.

SRv6 services can utilize either remote or local SIDs, depending on service requirements and deployment models. The integration with the writable SID Library (W-LIB) further enhances service delivery by efficiently managing SID resources and enabling dynamic service provisioning.

These SRv6 services are supported:

- L2 and L3 services with remote SIDs from W-LIB

- L3 services with local SIDs from W-LIB

These capabilities enable operators to deliver versatile and scalable network services using SRv6, leveraging both remote and local SIDs managed by W-LIB.

L2 and L3 services with remote SIDs from W-LIB

An L2 and L3 service with remote SIDs from Wide Local ID Block (W-LIB) is a SRv6 feature that

- uses SRv6 to enable a headend node to receive and install remote Segment Identifiers (SIDs) with wide (32-bit) functions
- supports both Layer and Layer service steering through these remote SIDs, and
- facilitates precise packet forwarding and service function chaining by embedding service SIDs into packet headers to steer traffic along specific paths including the intended service functions

This capability is enabled by default and allows the source node to insert a service SID into the packet header, which uniquely identifies a specific service function and directs the packet through the network accordingly.

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
SRv6 Services: L2 and L3 Services with Remote SIDs from Wide Local ID Block	Release 25.1.1	Introduced in this release on: Fixed Systems (8700 [ASIC: K100], 8010 [ASIC: A100]) This feature is now supported on: • 8712-MOD-M • 8011-4G24Y4H-I
SRv6 Services: L2 and L3 Services with Remote SIDs from Wide Local ID Block	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Feature Name	Release Information	Feature Description
SRv6 Services: L2 and L3 Services with Remote SIDs from Wide Local ID Block	Release 7.9.1	This feature enables an SRv6 headend node to receive and install remote SIDs with Wide (32-bit) functions (Remote W-LIB). The Remote W-LIB is supported for Layer 3 (VPN/BGP global) and Layer 2 EVPN services (ELINE/ELAN). This capability is enabled by default.

How the L2 and L3 services using remote SIDs from W-LIB work

Summary

The SRv6 headend node, source node, and receiver are the key components involved in SRv6 services with remote SIDs from W-LIB. Together, these components manage the identification, insertion, signaling (through transposition and SRv6 SID Structure Sub-Sub-TLV), and reassembly of SRv6 Service SIDs. This collaborative effort ensures precise Layer 2 and Layer 3 traffic steering for service function chaining and optimizes the compression of service prefix NLRIs in BGP update messages.

Workflow

These stages describe how the L2 and L3 services using remote SIDs from W-LIB work.

1. The source node inserts an SRv6 Service SID into the packet header to identify and steer the packet through the intended service function path.
2. The headend node signals the Service SID by transposing variable parts of the SRv6 SID value (function, argument, or both) into existing label fields, which optimizes compression of service prefix NLRIs in BGP updates.
3. The SRv6 SID Structure Sub-Sub-TLV (SSTLV) carries appropriate length fields that enable the receiver to accurately reassemble the split SRv6 Service SID parts. The Transposition Offset specifies the bit position, and the Transposition Length specifies how many bits are extracted from the SRv6 SID and placed into the high-order bits of the label field.
4. The receiver reassembles the split SRv6 Service SID parts based on the SSTLV information.
5. The headend node steers the packet along the specific path that includes the required service function, using the reassembled Service SID.

Remote W-LIB uSID structure with SRv6 SID SSTLV

A remote W-LIB uSID is defined by specific parameters within the SRv6 SID SSTLV that describe the composition and transposition of the SID value and label.

The example uSID fcbb:bb00:0200:fff0:0001:: is characterized as follows:

- Block length (BL) of 32 bits = fcbb:bb00
- Node length (NL) of 16 bits = 0200
- Function length (FL) of 32 bits = fff0:0001

- Argument length (AL) of 0
- Transposition length (TPOS len) of 16 bits = 0001
- Transposition offset (TPOS offset) of 64 bits = fcbb:bb00:0200:fff0:

From these parameters:

- The SID value is constructed as fcbb:bb00:0200:fff0::, which includes the block, node, and function parts without the transposed argument.
- The Label value extracted through transposition is 0x0001, representing the transposed bits from the original SID.

Interpret BGP VPNv4 route table output for a prefix learned from multiple egress PEs

Understand the details of BGP paths for a VPNv4 prefix learned from three egress Provider Edge (PE) routers

This task helps network engineers analyze BGP route entries, including path attributes and Segment Routing (SR) information, to troubleshoot or verify VPNv4 routing

Procedure

View the BGP route table for a VPNv4 prefix learned from three egress PEs.

- BGP Path 1 from next-hop 7::1 and a 32-bit uDT4 function (0xfff0 4002) allocated from W-LIB
- BGP Path 2 from next-hop 9::1 and a 16-bit uDT4 function (0x4002) allocated from LIB
- BGP Path 3 from next-hop 8::1 and a 16-bit uDT4 function (0x4002) allocated from LIB

The example shows output of a BGP route table for a VPNv4 prefix learned from three egress PEs:

Example:

```
Router# show bgp vpnv4 unicast rd 100:2 2.2.0.1/32 detail

BGP routing table entry for 2.2.0.1/32, Route Distinguisher: 100:2
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker               5314         5314
    Flags: 0x20061292+0x00060000; multipath; backup available;
Last Modified: Jan 20 14:37:59.189 for 00:00:19
Paths: (3 available, best #1)
  Not advertised to any peer
 Path #1: Received by speaker 0
  Flags: 0x2000000085070005+0x00, import: 0x39f
  Not advertised to any peer
  Local
    7::1 (metric 20) from 2::1 (192.0.0.1), if-handle 0x00000000
      Received Label 0x40020
      Origin IGP, localpref 150, valid, internal, best, group-best, multipath, import-candidate,
imported
      Received Path ID 1, Local Path ID 1, version 5314
      Extended community: RT:100:2
      Originator: 192.0.0.1, Cluster list: 2.0.0.1
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SubTLV:
```

```
      T:1(Sid information), Sid:fccc:cc00:7001:fff0::, F:0x00, R2:0x00, Behavior:63, R3:0x00, SS-TLV
 Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,32,0], Tpose-len:16, Tpose-offset:64
     Source AFI: VPNv4 Unicast, Source VRF: VRF_2, Source Route Distinguisher: 100:2
  Path #2: Received by speaker 0
  Flags: 0x2000000084060005+0x00, import: 0x096
  Not advertised to any peer
  Local
    9::1 (metric 20) from 2::1 (192.0.0.3), if-handle 0x00000000
      Received Label 0x40020
     Origin IGP, localpref 100, valid, internal, backup(protect multipath), add-path, import-candidate,
 imported
      Received Path ID 2, Local Path ID 5, version 5314
      Extended community: RT:100:2
      Originator: 192.0.0.3, Cluster list: 2.0.0.1
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SubTLV:
        T:1(Sid information), Sid:fccc:cc00:9001::, F:0x00, R2:0x00, Behavior:63, R3:0x00, SS-TLV
Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:16, Tpose-offset:48
     Source AFI: VPNv4 Unicast, Source VRF: VRF_2, Source Route Distinguisher: 100:2
  Path #3: Received by speaker 0
  Flags: 0x2000000084070005+0x00, import: 0x296
  Not advertised to any peer
  Local
    8::1 (metric 20) from 2::1 (192.0.0.2), if-handle 0x00000000
      Received Label 0x40020
      Origin IGP, localpref 150, valid, internal, multipath, backup, add-path, import-candidate,
imported
      Received Path ID 3, Local Path ID 4, version 5314
      Extended community: RT:100:2
      Originator: 192.0.0.2, Cluster list: 2.0.0.1
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SubTLV:
        T:1(Sid information), Sid:fccc:cc00:8001::, F:0x00, R2:0x00, Behavior:63, R3:0x00, SS-TLV
Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:16, Tpose-offset:48
     Source AFI: VPNv4 Unicast, Source VRF: VRF_2, Source Route Distinguisher: 100:2
```

Note the following fields in the output:

- Function length of 16 bits for LIB and 32 bits for W-LIB
- Transposition offset (Tpose-offset) value of 48 bits for LIB and 64 bits for W-LIB
- Transposition length (Tpose-len) value of 16 bits for LIB/W-LIB

L3 services with local SIDs from W-LIB

L3 services with local SIDs from W-LIB are network services that

- use SRv6 service SIDs to identify specific service functions
- steer packets along defined paths including those service functions, and

- enable flexible and efficient service delivery by allocating uSIDs from the W-LIB space.

Table 3: Feature History Table

Feature Name	Release	Description
SRv6-Services: L3 Services with Local SIDs from W-LIB	Release 25.1.1	Introduced in this release on: Fixed Systems (8700 [ASIC: K100], 8010 [ASIC: A100]) This feature is now supported on: • 8712-MOD-M • 8011-4G24Y4H-I
SRv6-Services: L3 Services with Local SIDs from W-LIB	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
SRv6-Services: L3 Services with Local SIDs from W-LIB	Release 7.11.1	This feature enables an SRv6 headend node to allocate and advertise local SIDs with Wide (32-bit) functions (Local W-LIB). The headend router utilizes the local W-LIB functionality to define and implement SR policies using SRv6 SIDs. The Local W-LIB is supported for Layer 3 (VPNv4/VPNv6/BGPv4/BGPv6 global) services. This feature introduces the **usid allocation wide-local-id-block** command.

SRv6 service SID and BGP uSID allocation

An SRv6 service SID uniquely identifies a specific service function within a network. The source node inserts this service SID into the packet header to steer the packet along a defined path that includes the targeted service function. This mechanism provides enhanced flexibility and control over packet processing, enabling efficient service delivery across the network.

Key points regarding uSID allocation and BGP transposition include:

- By default, BGP instructs the SID-Manager to allocate uSIDs from the LIB space only. When enabled, BGP can enforce uSID allocation from the W-LIB space.

- For VPN services, BGP performs transposition of the service SID into the label part of the NLRI as specified in IETF RFC 9252. In the LIB implementation, the 16-bit function is transposed to the NLRI label field.
- In the W-LIB implementation, BGP transposes the last 16 bits of the 32-bit W-LIB function to the NLRI label field for VPNv4 and VPNv6 routes.
- There is no transposition for BGPv4/BGPv6 global routes.

Support for Cisco Silicon One ASICs

- This feature is supported on Cisco 8000 Series Routers and Line Cards with Cisco Silicon One Q200 and P100 ASICs.
- This feature is not supported on Cisco 8000 Series Routers and Line Cards with Cisco Silicon One Q100 ASICs.

Configure L3 services with local SIDs from W-LIB

Enable the allocation and advertisement of an SRv6 service SID using the wide-local-id-block (W-LIB) mode to support L3 services

The W-LIB allocation mode applies precedence rules at various configuration levels within BGP to control how uSIDs are allocated and advertised.

Procedure

Step 1 Apply W-LIB uSID allocation globally under BGP

Example:

```
router bgp 1
  segment-routing srv6
    usid allocation wide-local-id-block
  !
```

Step 2 Apply W-LIB uSID allocation at the IPv4 and IPv6 address family levels under BGP.

Example:

```
router bgp 1
  address-family ipv4 unicast
    segment-routing srv6
      usid allocation wide-local-id-block
  !
  address-family ipv6 unicast
    segment-routing srv6
      usid allocation wide-local-id-block
```

Step 3 Apply W-LIB uSID allocation for all VPNv4 and VPNv6 address families.

Example:

```
router bgp 1
  address-family ipv4 unicast
    segment-routing srv6
      usid allocation wide-local-id-block
   !
```

```
  address-family ipv6 unicast
    segment-routing srv6
      usid allocation wide-local-id-block
```

Step 4 Apply W-LIB uSID allocation at the VRF level for IPv4 and IPv6 address families.

Example:

```
router bgp 1
  address-family vpnv4 unicast
    vrf all
      segment-routing srv6
        usid allocation wide-local-id-block
    !
  address-family vpnv6 unicast
    vrf all
      segment-routing srv6
        usid allocation wide-local-id-block
```

Step 5 Use the show commands to verify the W-LIB uSID allocation.

a) Verify the W-LIB uSID allocation.

Example:

```
RP/0/0/CPU0:PE1# show bgp ipv4 unicast process

BGP Process Information:
BGP is operating in STANDALONE mode
Autonomous System number format: ASPLAIN
Autonomous System: 100
Router ID: 192.168.0.1
Default Cluster ID: 192.168.0.1
Active Cluster IDs:  192.168.0.1
Fast external fallover enabled
Platform Loadbalance paths max: 16
Platform RLIMIT max: 2147483648 bytes
Maximum limit for BMP buffer size: 409 MB
Default value for BMP buffer size: 307 MB
Current limit for BMP buffer size: 307 MB
Current utilization of BMP buffer limit: 0 B
Neighbor logging is enabled
Enforce first AS enabled
AS Path multipath-relax is enabled
Use SR-Policy admin/metric of color-extcomm Nexthop during path comparison: disabled
Default local preference: 100
Default keepalive: 60
Graceful restart enabled
Restart time: 120
Stale path timeout time: 360
RIB purge timeout time: 600
Non-stop routing is enabled
ExtComm Color Nexthop validation: RIB

Update delay: 120
Generic scan interval: 15
Configured Segment-routing Local Block: [0, 0]
In use Segment-routing Local Block: [15000, 15999]
Platform support mix of sr-policy and native nexthop: No
Segment Routing SRv6 Locator Name: LOC2
Segment Routing SRv6 uSID WLIB allocation: Enforced

Address family: IPv4 Unicast
Dampening is enabled
```

```
Client reflection is enabled in global config
Dynamic MED is Disabled
Dynamic MED interval : 10 minutes
Dynamic MED Timer : Running, will expire in 342 seconds
Dynamic MED Periodic Timer : Running, will expire in 42 seconds
Scan interval: 60
Total prefixes scanned: 42
Prefixes scanned per segment: 100000
Number of scan segments: 1
Nexthop resolution minimum prefix-length: 0 (not configured)
IPv6 Nexthop resolution minimum prefix-length: 0 (not configured)
Main Table Version: 44
Table version synced to RIB: 44
Table version acked by RIB: 44
IGP notification: IGPs notified
RIB has converged: version 0
RIB table prefix-limit reached ?  [No], version 0
Permanent Network Unconfigured
Segment Routing SRv6 Alloc Mode: 0
Segment Routing SRv6 uSID WLIB allocation: Enforced


RP/0/0/CPU0:PE1# show bgp vrf all ipv4 unicast process

VRF: foo
-------

BGP Process Information: VRF foo
BGP Route Distinguisher: 23:1

BGP is operating in STANDALONE mode
Autonomous System number format: ASPLAIN
Autonomous System: 100
Router ID: 192.168.0.1
Default Cluster ID: 192.168.0.1
Active Cluster IDs:  192.168.0.1
Fast external fallover enabled
Platform Loadbalance paths max: 16
Platform RLIMIT max: 2147483648 bytes
Maximum limit for BMP buffer size: 409 MB
Default value for BMP buffer size: 307 MB
Current limit for BMP buffer size: 307 MB
Current utilization of BMP buffer limit: 0 B
Neighbor logging is enabled
Enforce first AS enabled
iBGP to IGP redistribution enabled
AS Path multipath-relax is enabled
Use SR-Policy admin/metric of color-extcomm Nexthop during path comparison: disabled
Default local preference: 100
Default keepalive: 60
Graceful restart enabled
Restart time: 120
Stale path timeout time: 360
RIB purge timeout time: 600
Non-stop routing is enabled
ExtComm Color Nexthop validation: RIB

Update delay: 120
Generic scan interval: 15
Configured Segment-routing Local Block: [0, 0]
In use Segment-routing Local Block: [15000, 15999]
Platform support mix of sr-policy and native nexthop: No
Segment Routing SRv6 Locator Name: LOC2 (WLIB allocation enforced)
Segment Routing SRv6 uSID WLIB allocation: Enforced
```

```
VRF foo Address family: IPv4 Unicast
Dampening is enabled
Client reflection is not enabled in global config
Dynamic MED is Disabled
Dynamic MED interval : 10 minutes
Dynamic MED Timer : Not Running
Dynamic MED Periodic Timer : Not Running
Scan interval: 60
Total prefixes scanned: 85
Prefixes scanned per segment: 100000
Number of scan segments: 1
Nexthop resolution minimum prefix-length: 0 (not configured)
IPv6 Nexthop resolution minimum prefix-length: 0 (not configured)
Main Table Version: 152
Table version synced to RIB: 152
Table version acked by RIB: 152
IGP notification: IGPs notified
RIB has converged: version 1
RIB table prefix-limit reached ?  [No], version 0
Permanent Network Unconfigured
Segment Routing SRv6 uSID WLIB allocation: Enforced
```

b) Verify the advertised SRv6 W-LIB uSID for the default VRF.

Example:

```
RP/0/0/CPU0:PE1# show bgp ipv4 unicast 192.168.4.1/32

BGP routing table entry for 192.168.4.1/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                419          419
    SRv6-VPN SID: fccc:cccc:a:fff0:4::/80
Last Modified: Apr  3 10:35:41.000 for 136y10w
Paths: (1 available, best #1)
  Advertised IPv4 Unicast paths to peers (in unique update groups):
    192::4
  Path #1: Received by speaker 0
  Advertised IPv4 Unicast paths to peers (in unique update groups):
    192::4
  Local
    0.0.0.0 from 0.0.0.0 (192.168.0.1)
      Origin incomplete, metric 0, localpref 100, weight 32768, valid, redistributed, best,
group-best
      Received Path ID 0, Local Path ID 1, version 419
```

c) Verify the advertised SRv6 W-LIB uSID for a specific VRF (foo).

Example:

```
RP/0/0/CPU0:PE1# show bgp vrf foo 192.168.7.1/32

BGP routing table entry for 192.168.7.1/32, Route Distinguisher: 23:1
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                439          439
    SRv6-VPN SID: fccc:cccc:a:fff0:4::/80
Last Modified: Apr  3 10:31:00.000 for 00:00:44
Paths: (1 available, best #1)
  Advertised to PE peers (in unique update groups):
    192::4
  Advertised to CE peers (in unique update groups):
```

```
    10.10.10.2
  Path #1: Received by speaker 0
  Advertised to PE peers (in unique update groups):
    192::4
  Advertised to CE peers (in unique update groups):
    10.10.10.2
  Local
    0.0.0.0 from 0.0.0.0 (192.168.0.1)
      Origin incomplete, metric 0, localpref 100, weight 32768, valid, redistributed, best,
group-best, import-candidate
      Received Path ID 0, Local Path ID 1, version 439
      Extended community: RT:23:23
```

Static SRv6 pseudowire

Static SRv6 pseudowire (PW) gives the flexibility to

- configure single-homing static SRv6 pseudowire between two Provider Edge (PE) routers, in an SRv6 core plane, and
- extends Virtual Private Wire Service (VPWS) capabilities by incorporating SRv6.

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
Static SRv6 pseudowire	Release 25.1.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100, Q200] (select variants only*); Modular Systems (8800 [LC ASIC: P100]); Centralized Systems (8600 [ASIC: Q200]) You can now create a virtual connection between two endpoints over an IPv6 network using SRv6. Unlike dynamic pseudowires, which rely on signaling protocols to establish and maintain connections, static pseudowires are manually configured and do not require a control plane for setup. This enhancement allows operators to extend the existing Virtual Private Wire Service (VPWS) capabilities by incorporating SRv6, providing improved flexibility and scalability for service providers. * This feature is supported on: • 8201-32FH

Key aspects of SRv6 static pseudowire

This feature enhances the existing Virtual Private Wire Service (VPWS) by incorporating SRv6 capabilities, which previously supported only MPLS.

This enhancement enables the configuration and management of static SRv6 pseudowires, thereby providing enhanced flexibility and scalability for service providers. Additionally, it addresses the specific requirements of customers who require static configuration for their network services.

- Static configuration: Unlike dynamic pseudowires that rely on signaling protocols like LDP or BGP, static pseudowires are manually configured. This allows for more control and customization in specific network scenarios.
- Pseudowire (PW): A pseudowire is a mechanism that emulates the properties of a traditional telecommunications circuit over a packet-switched network. It allows for the transport of Layer 2 frames over an IP/MPLS network.

The benefits of having an option to configure pseudowire in a static SRv6 core plane gives businesses an edge over their competitors, as listed.

- Enhanced flexibility: By using SRv6, service providers can define more granular and flexible paths for their pseudowires, improving network efficiency and performance.
- Scalability: SRv6 allows for a scalable solution that can handle a large number of pseudowires without the need for complex BGP signaling protocols.
- Interoperability: The feature is designed to work across various platforms and planes making it versatile for different network environments.

Configure Static SRv6 pseudo-wire

The purpose of this task is to configure static SRv6 pseudo-wires. To configure this, the network operators must define the local and remote SRv6 SIDs for the pseudo-wire endpoints. This involves specifying the segment routing configuration and ensuring that the pseudo-wire is correctly mapped to the desired network path.

Procedure

Step 1 Define the locator.

Example:

```
Router(config)#segment-routing
Router(config-sr)#srv6
Router(config-srv6)#formats
Router(config-srv6-fmts)#format usid-f3216
Router(config-srv6-fmts)#usid local-id-block explicit start 0xee00 (default start is 0xfe00 - optional
 required if you if more than 256 )
Router(config-srv6-fmts)#exit

Router(config-srv6)#encapsulation
Router(config-srv6)#source-address 1::1
Router(config-srv6)traffic-class propagate

Router(config-srv6)#locators
Router(config-srv6-locators)#locator locator0
Router(config-srv6-locators)#micro-segment behavior unode psp-usd
Router(config-srv6-locators)#prefix fccc:cc00:1::/48
```

Step 2 Define the Sub-interface.

Example:

```
interface Bundle-Ether101.40010001 l2transport
 encapsulation dot1q 1
 rewrite ingress tag pop 1 symmetric
!
```

Step 3 Define the xconnect group.

Example:

```
Router(config)#l2vpn
Router(config-l2vpn)#xconnect group xg4001
Router(config-l2vpn-xc)#p2p vpws-400100150
Router(config-l2vpn-xc-p2p)#interface Bundle-Ether101
Router(config-l2vpn-xc-p2p)#neighbor segment-routing srv6 static local fccc:cc00:1:ee96:: remote
fccc:cc00:2:ee96::
```

Note
The **remote fccc:cc00:2:ee96::** SID is the remote device SID. You must configure this on both ends of the PE. However the local and remote SIDs interrchange

Step 4 Run the following show commands to view the defined locator, evpn, L2vpn, xconnect and evpn internal ID details.

Example:

```
Router#show segment-routing srv6 locator locator0 sid  fccc:cc00:1:ee01:: detail

SID                         Behavior          Context                          Owner
 State  RW
--------------------------  ----------------  -------------------------------  ------------------
 -----  --
fccc:cc00:1:ee01::          uDX2              pw_id:3221225475                 l2vpn_srv6
 InUse  Y
  SID Function: 0xee01
  SID context: { static pw_id=3221225475 }
  Locator: 'locator0'
  Allocation type: Explicit
  Created: Dec 16 15:38:33.852 (00:05:19 ago)


Router# show evpn segment-routing srv6 detail

Configured default locator: None
Configured default SID Function Length: 16 bits
EVIs with unknown locator config: 0
VPWS with unknown locator config: 0
Global SID Function Length: 16 bits
No SRv6 locators in use

Router#show l2vpn xconnect group xg4001 xc-name vpws-40010001 det

Group xg4001, XC vpws-40010001, state is up; Interworking none
Decoupled mode: Disabled
  AC: Bundle-Ether101.40010001, state is up
    Type VLAN; Num Ranges: 1
    Rewrite Tags: []
    VLAN ranges: [1, 1]
    MTU 9186; XC ID 0xc0000003; interworking none
    Statistics:
```

```
      packets: received 7109476, sent 5069183
      bytes: received 910012144, sent 648854934
      drops: illegal VLAN 0, illegal length 0
  PW: neighbor ::ffff:10.0.0.4, PW ID 2684354565, state is up
    PW class not set, XC ID 0xc0000003
    Encapsulation SRv6, protocol none
    PW type Ethernet, control word unknown, interworking none
    PW backup disable delay 0 sec

    Ignore MTU mismatch: Disabled
    Transmit MTU zero: Disabled
    Reachability: Up
    Nexthop type: Internal ID ::ffff:10.0.0.4

      SRv6              Local                       Remote
      ----------------  --------------------------- --------------------------
      uDX2              fccc:cc00:1:ee01::          fccc:cc00:2:ee01::
      AC ID             0                           0
      MTU               9200                        9200
      Locator           locator0                    N/A
      Locator Resolved  Yes                         N/A
      SRv6 Headend      H.Encaps.L2.Red             N/A
    Statistics:
      packets: received 5069183, sent 7109476
      bytes: received 648854934, sent 910012144
```

```
Router#show evpn internal-id

VPN-ID     Encap  Ethernet Segment Id         EtherTag     Internal ID
---------- ------ -------------------------- ----------   -----------
0          SRv6   fccc:cc00:2:ee01::          13421820     ::ffff:10.0.0.4
   Summary pathlist (ID 0x000000000000001a):
 0x05000005     fccc:cc00:2:ee01::                         fccc:cc00:2:ee01::

0          SRv6   fccc:cc00:2:ee02::          13421820     ::ffff:10.0.0.2
   Summary pathlist (ID 0x000000000000000a):
 0x05000002     fccc:cc00:2:ee02::                         fccc:cc00:2:ee02::
```

```
Router#show evpn internal-id  detail

VPN-ID     Encap  Ethernet Segment Id         EtherTag     Internal ID
---------- ------ -------------------------- ----------   -----------
0          SRv6   fccc:cc00:2:ee01::          13421820     ::ffff:10.0.0.4
   Path resolved: TRUE
   Path Internal ID: ::ffff:10.0.0.4
    IP-Tunnel (ID:0x0000000000000019)
                fccc:cc00:2:ee01::                         fccc:cc00:2:ee01::
                  Path Version:1, Originating PE:::
   Summary pathlist (ID 0x000000000000001a):
 0x05000005     fccc:cc00:2:ee01::                         fccc:cc00:2:ee01::

0          SRv6   fccc:cc00:2:ee02::          13421820     ::ffff:10.0.0.2
   Path resolved: TRUE
   Path Internal ID: ::ffff:10.0.0.2
    IP-Tunnel (ID:0x0000000000000009)
                fccc:cc00:2:ee02::                         fccc:cc00:2:ee02::
                  Path Version:1, Originating PE:::
   Summary pathlist (ID 0x000000000000000a):
 0x05000002     fccc:cc00:2:ee02::                         fccc:cc00:2:ee02::
```