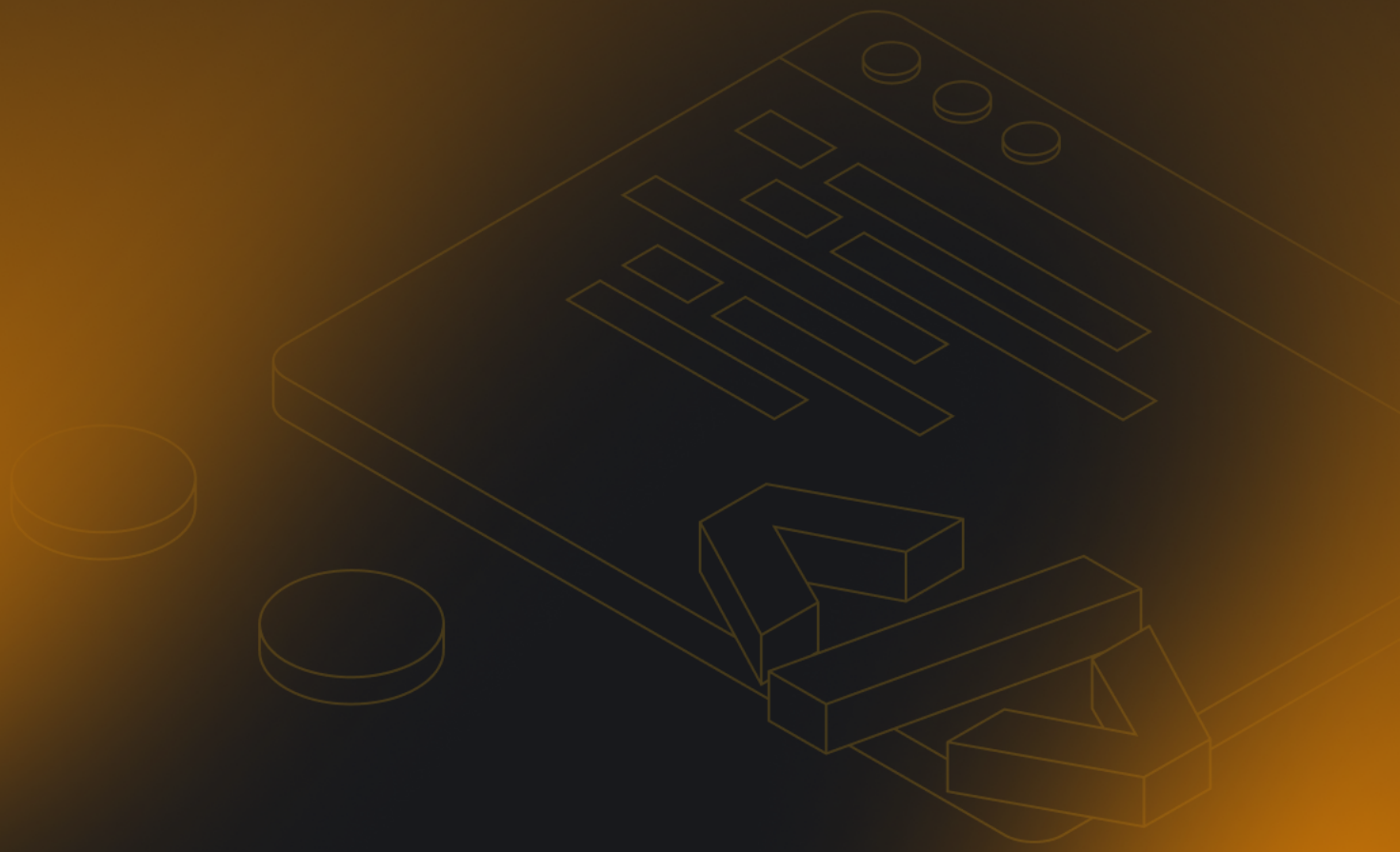
VyOS Networks



Deployment Guide | Technical Doc

SRV6 - DEPLOYMENT L3VPN (VPNV6/VPNV4)

July 2025

Index:

Introduction

This document serves as a contribution to the VyOS Community and provides a comprehensive guide for configuring Segment Routing over IPv6 (SRv6) on VyOS. It specifically demonstrates the setup of L3VPN services using an IPv6 overlay deployed over an IS-IS unnumbered network. By leveraging IPv6 link-local addresses for dynamic neighbor discovery, this approach eliminates the need for manual IPv6 underlay configuration. The result is a more scalable and automated design, ideally suited for large service provider backbone networks.

Scenario | Topology

This lab is based on a straightforward service provider topology designed to simulate a typical MPLS Layer 3 VPN scenario using Segment Routing over IPv6 (SRv6). The setup includes two Provider Edge (PE) routers and two remote Customer Edge (CE) routers, with each CE connected to a corresponding PE router across an IP backbone.

Core Network Configuration (Ps Routers)

The IP backbone, composed of core P routers, utilizes IS-IS with unnumbered interfaces as the Interior Gateway Protocol (IGP). This configuration facilitates the distribution of Segment Routing IPv6 (SRv6) SIDs across the IPv6 network. SRv6 relies on link-local IPv6 addresses to establish neighbor adjacencies and assign routing identifiers.

Provider Edge Communication (PEs Routers)

An iBGP session is established between the PE routers, supporting both IPv4 and IPv6 VPN address families. This dual-stack configuration ensures that VPNv4 and VPNv6 routes can be exchanged between PEs effectively.

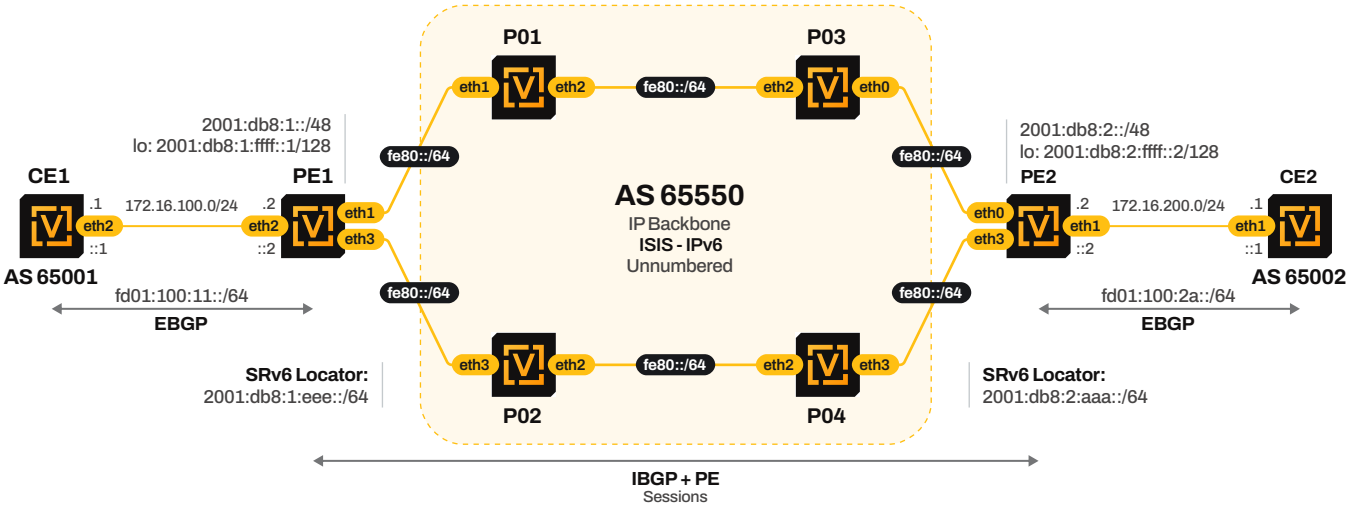
Customer Edge Connectivity (CEs Routers)

An iBGP session is established between the PE routers, supporting both IPv4 and IPv6 VPN address families. This dual-stack configuration ensures that VPNv4 and VPNv6 routes can be exchanged between PEs effectively.

- **CE1** is connected to **PE1** via a standard dual-stack Ethernet link, supporting both IPv4 and IPv6.
- **CE2** is similarly connected to **PE2** with a standard dual-stack Ethernet link.

This architecture enables seamless route exchange between the customer sites using SRv6 as the transport mechanism, without requiring native end-to-end IPv4 connectivity across the backbone. It exemplifies a modernized alternative to traditional MPLS L3VPN deployments, where the IPv6 infrastructure carries VPNv4/V6 traffic using SRv6 encapsulation.

SRv6 Architecture



Configurations and Deployment

SRv6 L3VPN deployment

As seen in the diagram below , we are deploying a dedicated VRF instance for a customer, named **customer-1**. This VRF is configured with a dual-stack IP addressing scheme (both IPv4 and IPv6), enabling communication for remote site services over both protocols. Despite the underlying ISP backbone being IPv6-only, this configuration ensures full support for dual-stack connectivity between sites:

Let's see a simple configuration of the IP Backbone.

IP Backbone - Provider

Provider Routers

ISIS unnumbered and basic ipv6 configuration:

P01:

```
set interfaces ethernet eth1 description 'PE1'
set interfaces ethernet eth2 description 'P03'

# ISIS unnumbered:

set protocols isis interface eth1
set protocols isis interface eth2
set protocols isis level 'level-2'
set protocols isis net '49.0000.0000.1111.1111.00'

# hostname :
set system host-name 'P01'

# similar configuration to P02,P03 and P04.
```

Provider Edges Routers

We are going to start with basic configuration to enable ISIS as IGP and SRv6:

SRv6:

To begin, we enable the interfaces that will receive SRv6 traffic—specifically, our provider-facing network interfaces. Additionally, we must define at least one prefix to serve as a base for allocating Segment IDs (SIDs), which are essential for SRv6 path identification.

PE1:

```
# basic SRv6 configuration, SID locator

set protocols segment-routing interface eth1 srv6
set protocols segment-routing interface eth3 srv6
set protocols segment-routing srv6 locator L3VPN behavior-usid
set protocols segment-routing srv6 locator L3VPN prefix '2001:db8:1:eee::/64'
```

PE2:

```
# basic SRv6 configuration, SID locator

set protocols segment-routing interface eth0 srv6
set protocols segment-routing interface eth3 srv6
set protocols segment-routing srv6 locator L3VPN behavior-usid
set protocols segment-routing srv6 locator L3VPN prefix '2001:db8:2:aaa::/64'
```

ISIS IGP - Unnumbered

IGP ISIS unnumbered and srv6 capability:

```
PE1:

set interfaces loopback lo address '2001:db8:1:ffff::1/128'

set interfaces ethernet eth1 description 'P01'
set interfaces ethernet eth3 description 'P02'

set protocols isis interface eth1
set protocols isis interface eth3
set protocols isis interface lo
set protocols isis level 'level-2'
set protocols isis net '49.0000.0000.0000.0001.00'

# enable SRv6 on ISIS to redistribute  SID prefix

set protocols isis segment-routing srv6 locator L3VPN

PE2:

set interfaces loopback lo address '2001:db8:2:ffff::2/128'
set interfaces ethernet eth0 description 'P03'
set interfaces ethernet eth3 description 'P04'
```

```
set protocols isis interface eth0
set protocols isis interface eth3
set protocols isis interface lo
set protocols isis level 'level-2'
set protocols isis net '49.0000.0000.0000.0002.00'

# enable SRv6 on ISIS to redistribute  SID prefix

set protocols isis segment-routing srv6 locator L3VPN
```

BGP overlay configuration

In this step, we will configure a BGP relationship between two Provider Edge (PE) routers using IPv6 as the transport protocol. This setup enables the exchange of both IPv4 VPN and IPv6 VPN address families, allowing seamless distribution of IPv4 and IPv6 customer (VRF) routes between the PEs:

```
PE1:

set protocols bgp neighbor 2001:db8:2:ffff::2 description 'PE2'
set protocols bgp neighbor 2001:db8:2:ffff::2 peer-group 'SRv6'
set protocols bgp parameters router-id '10.0.0.1'
set protocols bgp peer-group SRv6 address-family ipv4-vpn nexthop-self
set protocols bgp peer-group SRv6 address-family ipv6-vpn nexthop-self
set protocols bgp peer-group SRv6 remote-as 'internal'
set protocols bgp peer-group SRv6 update-source 'lo'
set protocols bgp system-as '65550'

PE2:

set protocols bgp neighbor 2001:db8:1:ffff::1 description 'PE1'
set protocols bgp neighbor 2001:db8:1:ffff::1 peer-group 'SRv6'
set protocols bgp parameters router-id '10.0.0.2'
set protocols bgp peer-group SRv6 address-family ipv4-vpn nexthop-self
set protocols bgp peer-group SRv6 address-family ipv6-vpn nexthop-self
set protocols bgp peer-group SRv6 remote-as 'internal'
set protocols bgp peer-group SRv6 update-source 'lo'
set protocols bgp system-as '65550'
```

RFC 8950 introduces a mechanism that allows the use of an IPv6 address as the next-hop for IPv4 route advertisements. This capability is essential for enabling L3VPN functionality in Linux environments when dealing with customer IPv4 prefixes.

In this context, the IPv4 customer routes are advertised with an IPv6 next-hop, which corresponds to the egress Provider Edge (PE) router's IPv6 address. Additionally, these routes are associated with an IPv6 Segment Identifier (SID), allowing proper forwarding through the Segment Routing over IPv6 (SRv6) data plane. This integration ensures that IPv4 traffic can be encapsulated and transported effectively across an IPv6-based provider core.

```
# enable RFC8950 on BGP peer-group

PE1 / PE2 :

set protocols bgp peer-group SRv6 capability extended-nexthop
```

Configure SRv6 Locator that BGP should use to generate SIDs, it was previous configure on SRv6 :

```
PE1 / PE2 :

set protocols bgp srv6 locator 'L3VPN'
```

Overlay - customer VRFs

Our customer-specific VRF, **'customer-1'**, is deployed on both Provider Edge (PE) routers. This setup follows a traditional MPLS Layer 3 VPN configuration model. To ensure proper route exchange, we need to configure route distinction, as well as the import and export policies, between the **customer-1** and the global BGP IPv4/IPv6 VPN routing table. This enables seamless propagation of customer routes across the ISP backbone.

```
# VRF customer-1 VPNv4/VPNv6 address-family

# PE1:

set vrf name customer-1 protocols bgp address-family ipv4-unicast export vpn
set vrf name customer-1 protocols bgp address-family ipv4-unicast import vpn
set vrf name customer-1 protocols bgp address-family ipv4-unicast rd vpn export '10.0.0.1:100'
set vrf name customer-1 protocols bgp address-family ipv4-unicast redistribute connected
set vrf name customer-1 protocols bgp address-family ipv4-unicast route-target vpn export '65550:100'
set vrf name customer-1 protocols bgp address-family ipv4-unicast route-target vpn import '65550:100'
set vrf name customer-1 protocols bgp address-family ipv6-unicast export vpn
set vrf name customer-1 protocols bgp address-family ipv6-unicast import vpn
set vrf name customer-1 protocols bgp address-family ipv6-unicast rd vpn export '10.0.0.1:100'
set vrf name customer-1 protocols bgp address-family ipv6-unicast redistribute connected
set vrf name customer-1 protocols bgp address-family ipv6-unicast route-target vpn export '65550:100'
set vrf name customer-1 protocols bgp address-family ipv6-unicast route-target vpn import '65550:100'
set vrf name customer-1 protocols bgp parameters router-id '10.0.0.1'
set vrf name customer-1 protocols bgp system-as '65550'
set vrf name customer-1 table '100'

#PE2:

set vrf name customer-1 protocols bgp address-family ipv4-unicast export vpn
set vrf name customer-1 protocols bgp address-family ipv4-unicast import vpn
set vrf name customer-1 protocols bgp address-family ipv4-unicast rd vpn export '10.0.0.2:100'
set vrf name customer-1 protocols bgp address-family ipv4-unicast redistribute connected
set vrf name customer-1 protocols bgp address-family ipv4-unicast route-target vpn export '65550:100'
set vrf name customer-1 protocols bgp address-family ipv4-unicast route-target vpn import '65550:100'
set vrf name customer-1 protocols bgp address-family ipv6-unicast export vpn
set vrf name customer-1 protocols bgp address-family ipv6-unicast import vpn
set vrf name customer-1 protocols bgp address-family ipv6-unicast rd vpn export '10.0.0.2:100'
set vrf name customer-1 protocols bgp address-family ipv6-unicast redistribute connected
set vrf name customer-1 protocols bgp address-family ipv6-unicast route-target vpn export '65550:100'
set vrf name customer-1 protocols bgp address-family ipv6-unicast route-target vpn import '65550:100'
set vrf name customer-1 protocols bgp system-as '65550'
set vrf name customer-1 table '100'
```

SRv6 has been enabled for the VRF configuration. The Segment Identifier (SID) generation mode is set to per-vrf, ensuring that each VRF is assigned a unique SID. The SID allocation method is configured as auto, allowing the system to dynamically generate the SID:

```
PE1 / PE2 :

set vrf name customer-1 protocols bgp sid vpn per-vrf export 'auto'
```

The final step in configuring our Provider Edge (PE) routers involves defining the BGP neighbor on the Customer Edge (CE) device, which operates in dual-stack mode — supporting both IPv4-unicast and IPv6-unicast address families.

```
# PE1 CE - Configuration :

set interfaces ethernet eth2 address '172.16.100.2/24'
set interfaces ethernet eth2 address 'fd01:100:11::2/64'
set interfaces ethernet eth2 description 'CE1'
set interfaces ethernet eth2 vrf 'customer-1'

set vrf name customer-1 protocols bgp neighbor fd01:100:11::1 peer-group 'CE'
set vrf name customer-1 protocols bgp parameters router-id '10.0.0.1'
set vrf name customer-1 protocols bgp peer-group CE address-family ipv4-unicast
set vrf name customer-1 protocols bgp peer-group CE address-family ipv6-unicast
set vrf name customer-1 protocols bgp peer-group CE remote-as 'external

# PE2 CE - Configuration :

set interfaces ethernet eth1 address '172.16.200.2/24'
set interfaces ethernet eth1 address 'fd01:100:2a::2/64'
set interfaces ethernet eth1 description 'CE2'
set interfaces ethernet eth1 vrf 'customer-1'

set vrf name customer-1 protocols bgp neighbor fd01:100:2a::1 peer-group 'CE'
set vrf name customer-1 protocols bgp parameters router-id '10.0.0.2'
set vrf name customer-1 protocols bgp peer-group CE address-family ipv4-unicast
set vrf name customer-1 protocols bgp peer-group CE address-family ipv6-unicast
set vrf name customer-1 protocols bgp peer-group CE remote-as 'external'
```

Customer Edge

In this section, we will configure our Customer Edge (CE) devices and perform end-to-end testing of the solution. The configuration will support both IPv4 and IPv6 protocols, enabling a dual-stack setup to ensure comprehensive validation of the solution across both address families.

```
# CE1 configuration:

set interfaces ethernet eth2 address '172.16.100.1/24'
set interfaces ethernet eth2 address 'fd01:100:11::1/64'
set interfaces ethernet eth2 description 'PE2'

set protocols bgp address-family ipv4-unicast
set protocols bgp address-family ipv6-unicast
set protocols bgp neighbor fd01:100:11::2 address-family ipv4-unicast
set protocols bgp neighbor fd01:100:11::2 address-family ipv6-unicast
set protocols bgp neighbor fd01:100:11::2 remote-as 'external'
set protocols bgp system-as '65001'

#CE2 Configuration:
```

```
set interfaces ethernet eth1 address '172.16.200.1/24'
set interfaces ethernet eth1 address 'fd01:100:2a::1/64'
set interfaces ethernet eth1 description 'PE2'

set protocols bgp address-family ipv4-unicast
set protocols bgp address-family ipv6-unicast
set protocols bgp neighbor fd01:100:2a::2 address-family ipv4-unicast
set protocols bgp neighbor fd01:100:2a::2 address-family ipv6-unicast
set protocols bgp neighbor fd01:100:2a::2 remote-as 'external'
set protocols bgp system-as '65002'
```

Validation and Troubleshooting

SRv6 L3VPN validation:

1. Verify ISIS session are established:

```
vyos@PE1:~$ show isis neighbor
Area VyOS:
 System Id           Interface   L  State         Holdtime SNPA
 P01                 eth1        2  Up            29       5000.0002.0001
 P02                 eth3        2  Up            29       5000.0007.0003
```

2. Verify ISIS routes :

```
vyos@PE1# show ipv6 route isis
Codes: K - kernel route, C - connected, L - local, S - static,
       R - RIPng, O - OSPFv3, I - IS-IS, B - BGP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, t - Table-Direct,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

I>* 2001:db8:1:eee:2::/80 [115/0] is directly connected, eth1, seg6local End.X nh6 fe80::5200:ff:fe02:1,
weight 1, 01:25:10
I>* 2001:db8:1:eee:3::/80 [115/0] is directly connected, eth3, seg6local End.X nh6 fe80::5200:ff:fe07:3,
weight 1, 01:25:10
I>* 2001:db8:2:aaa::/64 [115/30] via fe80::5200:ff:fe02:1, eth1, weight 1, 01:23:21
  *                              via fe80::5200:ff:fe07:3, eth3, weight 1, 01:23:21
I>* 2001:db8:2:ffff::2/128 [115/40] via fe80::5200:ff:fe02:1, eth1, weight 1, 01:29:55
  *                                 via fe80::5200:ff:fe07:3, eth3, weight 1, 01:29:55
```

3. Verify that we reach the PE loopback via ISIS.

```
vyos@PE1:~$ ping 2001:db8:2:ffff::2 source-address 2001:db8:1:ffff::1
PING 2001:db8:2:ffff::2(2001:db8:2:ffff::2) from 2001:db8:1:ffff::1 : 56 data bytes
64 bytes from 2001:db8:2:ffff::2: icmp_seq=1 ttl=62 time=1.03 ms
64 bytes from 2001:db8:2:ffff::2: icmp_seq=2 ttl=62 time=1.48 ms
64 bytes from 2001:db8:2:ffff::2: icmp_seq=3 ttl=62 time=1.12 ms
^C
--- 2001:db8:2:ffff::2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.029/1.210/1.479/0.193 ms
```

4. Verify our BGP sessions are established

```
vyos@PE1:~$ show bgp summary

IPv4 VPN Summary:
BGP router identifier 10.0.0.1, local AS number 65550 VRF default vrf-id 0
BGP table version 0
RIB entries 3, using 384 bytes of memory
Peers 1, using 24 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor           V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:2:ffff::2 4      65550     18826     18826        3    0    0 01w6d01h            1        1 PE2

Total number of neighbors 1

IPv6 VPN Summary:
BGP router identifier 10.0.0.1, local AS number 65550 VRF default vrf-id 0
BGP table version 0
RIB entries 3, using 384 bytes of memory
Peers 1, using 24 KiB of memory
Peer groups 1, using 64 bytes of memory

State/PfxRcd   PfxSnt Desc
2001:db8:2:ffff::2 4      65550     18826     18826        3    0    0 01w6d01h            1        1 PE2

Total number of neighbors 1
```

5. Verify SRv6 SIDs:

```
vyos@PE1:~$ show ipv6 route bgp
Codes: K - kernel route, C - connected, L - local, S - static,
       R - RIPng, O - OSPFv3, I - IS-IS, B - BGP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, t - Table-Direct,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

B>* 2001:db8:1:eee:1::/128 [20/0] is directly connected, customer-1, seg6local End.DT46 table 100, weight
1, 01w6d19h

#linux command :

vyos@PE1:~$  ip -6 route show 2001:db8:1:eee:1::/128
2001:db8:1:eee:1:: nhid 23  encap seg6local action End.DT46 vrftable customer-1 dev customer-1 proto bgp
metric 20 pref medium
```

Note: An **End.DT46** Segment Identifier (SID) has been provisioned for customer-1 to enable advanced SRv6 endpoint functionality. For detailed information on SRv6 endpoint behaviors, refer to **RFC 8986.**

6. Let's see VPNv4/VPNv6 routes, we validate prefix in bpg table view:

VPNv4 prefix:

```
vyos@PE1:~$ show bgp ipv4 vpn
BGP table version is 3, local router ID is 10.0.0.1, vrf id 0
Default local pref 100, local AS 65550
Status codes:  s suppressed, d damped, h history, u unsorted, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 10.0.0.1:100
 *>  172.16.100.0/24  0.0.0.0@7<               0         32768 ?
    UN=0.0.0.0 EC{65550:100} label=16 sid=2001:db8:1:eee:: sid_structure=[40,24,16,0] type=bgp, subtype=5
Route Distinguisher: 10.0.0.2:100
 *>i 172.16.200.0/24  2001:db8:2:ffff::2
                                      0    100      0 ?
    UN=2001:db8:2:ffff::2 EC{65550:100} label=16 sid=2001:db8:2:aaa:: sid_structure=[40,24,16,0] type=bgp,
subtype=0

Displayed 2 routes and 2 total paths



vyos@PE1:~$ show bgp ipv4 vpn 172.16.200.0/24
BGP routing table entry for 10.0.0.2:100:172.16.200.0/24, version 1
not allocated
Paths: (1 available, best #1)
  Not advertised to any peer
  Local
    0.0.0.0 (metric 30) from 2001:db8:2:ffff::2 (10.0.0.2)
      Origin incomplete, metric 0, localpref 100, valid, internal, best (First path received)
      Extended Community: RT:65550:100
      Remote label: 16
      Remote SID: 2001:db8:2:aaa::
      Last update: Wed Jul  2 19:49:09 2025
```

VPNv6 prefix:

```
vyos@PE1:~$ show bgp ipv6 vpn
BGP table version is 3, local router ID is 10.0.0.1, vrf id 0
Default local pref 100, local AS 65550
Status codes:  s suppressed, d damped, h history, u unsorted, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 10.0.0.1:100
*>  fd01:100:11::/64 ::@7<                    0         32768 ?
    UN=:: EC{65550:100} label=16 sid=2001:db8:1:eee:: sid_structure=[40,24,16,0] type=bgp, subtype=5
Route Distinguisher: 10.0.0.2:100
*>i fd01:100:2a::/64 2001:db8:2:ffff::2

UN=2001:db8:2:ffff::2 EC{65550:100} label=16 sid=2001:db8:2:aaa:: sid_structure=[40,24,16,0] type=bgp,
subtype=0
```

```
Displayed 2 routes and 2 total paths

vyos@PE1:~$ show bgp ipv6 vpn fd01:100:2a::/64
BGP routing table entry for 10.0.0.2:100:fd01:100:2a::/64, version 1
not allocated
Paths: (1 available, best #1)
  Not advertised to any peer
  Local
    2001:db8:2:ffff::2 (metric 30) from 2001:db8:2:ffff::2 (10.0.0.2)
      Origin incomplete, metric 0, localpref 100, valid, internal, best (First path received)
      Extended Community: RT:65550
      Remote label: 16
      Remote SID: 2001:db8:2:aaa::
      Last update: Wed Jul  2 19:49:09 2025
```

7. Let's show vrf customer-1 route table, we get router from ipv4 and ipv6 address-family:

IPv4 :

```
vyos@PE1:~$ show ip route vrf customer-1
Codes: K - kernel route, C - connected, L - local, S - static,
       R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, t - Table-Direct,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF customer-1:
K>* 127.0.0.0/8 [0/0] is directly connected, customer-1, weight 1, 01w6d23h
C>* 172.16.100.0/24 is directly connected, eth2, weight 1, 01w6d23h
L>* 172.16.100.2/32 is directly connected, eth2, weight 1, 01w6d23h
B>  172.16.200.0/24 [200/0] via 2001:db8:2:ffff::2 (vrf default) (recursive), label 16, seg6
2001:db8:2:aaa:1::, weight 1, 01w6d22h
  *                           via fe80::5200:ff:fe02:1, eth1 (vrf default), label 16, seg6
2001:db8:2:aaa:1::, weight 1, 01w6d22h
  *                           via fe80::5200:ff:fe07:3, eth3 (vrf default), label 16, seg6
2001:db8:2:aaa:1::, weight 1, 01w6d22h
```

IPv6 :

```
vyos@PE1:~$ show ipv6 route vrf customer-1
Codes: K - kernel route, C - connected, L - local, S - static,
       R - RIPng, O - OSPFv3, I - IS-IS, B - BGP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, t - Table-Direct,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF customer-1:
K>* ::1/128 [0/256] is directly connected, customer-1, weight 1, 01w6d23h
C>* fd01:100:11::/64 is directly connected, eth2, weight 1, 01w6d23h
K * fd01:100:11::/64 [0/256] is directly connected, eth2, weight 1, 01w6d23h
L>* fd01:100:11::2/128 is directly connected, eth2, weight 1, 01w6d23h
B>  fd01:100:2a::/64 [200/0] via 2001:db8:2:ffff::2 (vrf default) (recursive), label 16, seg6
2001:db8:2:aaa:1::, weight 1, 01w6d22h
  *                            via fe80::5200:ff:fe02:1, eth1 (vrf default), label 16, seg6
2001:db8:2:aaa:1::, weight 1, 01w6d22h
```

```
  *                          via fe80::5200:ff:fe07:3, eth3 (vrf default), label 16, seg6
2001:db8:2:aaa:1::, weight 1, 01w6d22h
C>* fe80::/64 is directly connected, eth2, weight 1, 01w6d23h
```

Note: Before L3VPN traffic can be routed correctly, it is necessary to initiate traffic sourced from one Provider Edge (PE) router to the other. This exchange must occur within the associated VRF and specifically between interfaces that are members of the VRF on each PE. This initial communication is required to ensure proper forwarding behavior.

Eg:

```
vyos@PE1:~$ ping 172.16.200.2 source-address 172.16.100.2 vrf customer-1
PING 172.16.200.2 (172.16.200.2) from 172.16.100.2 : 56(84) bytes of data.
64 bytes from 172.16.200.2: icmp_seq=1 ttl=64 time=1.15 ms
64 bytes from 172.16.200.2: icmp_seq=2 ttl=64 time=1.47 ms
64 bytes from 172.16.200.2: icmp_seq=3 ttl=64 time=1.53 ms
√64 bytes from 172.16.200.2: icmp_seq=4 ttl=64 time=1.36 ms
^C
--- 172.16.200.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 1.147/1.375/1.529/0.146 ms
```

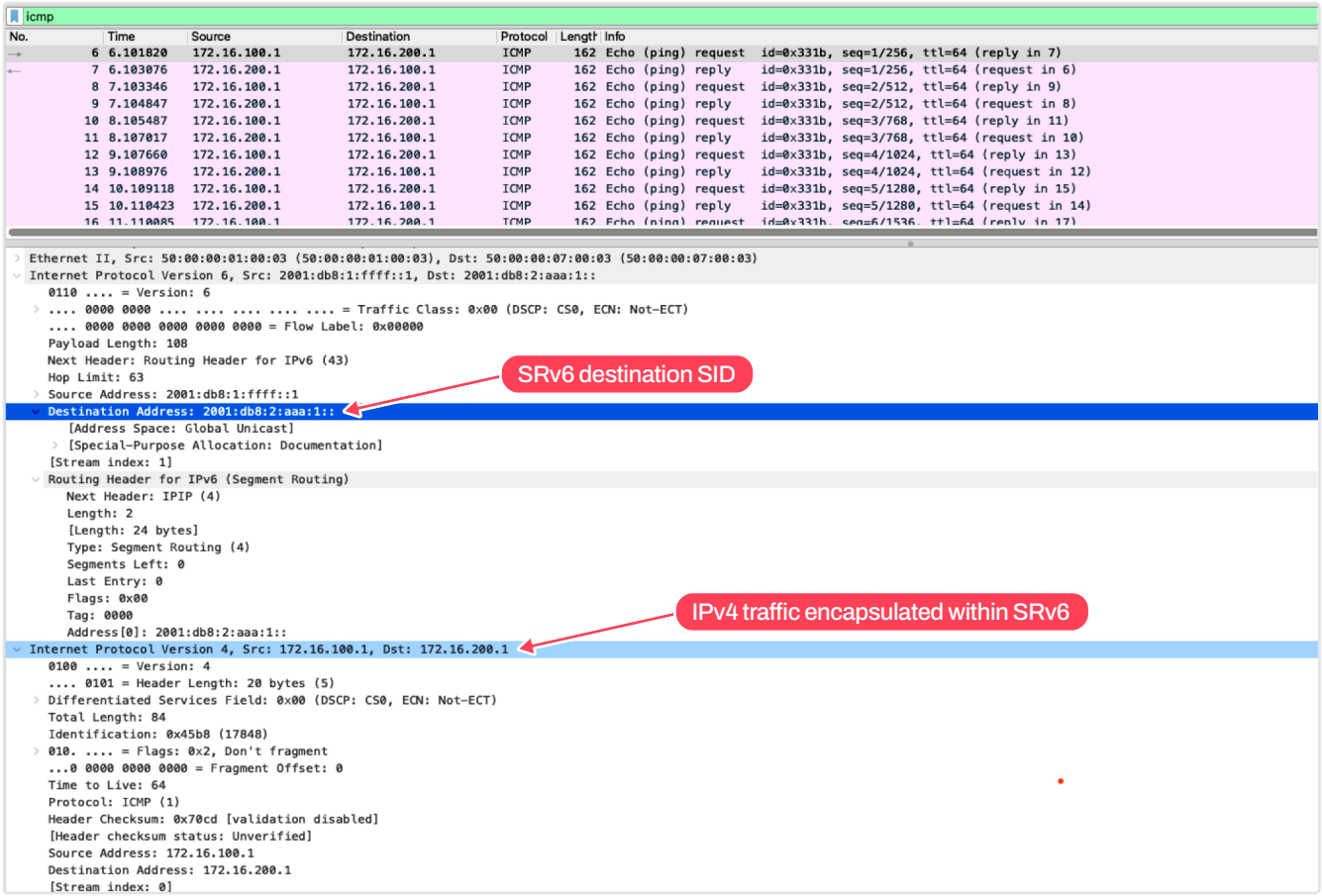
8. Finally, Let's try a ping test and capture traffic from CE-1 to CE-2:

IPv4:

```
vyos@CE1:~$ ping 172.16.200.1
PING 172.16.200.1 (172.16.200.1) 56(84) bytes of data.
64 bytes from 172.16.200.1: icmp_seq=1 ttl=63 time=1.68 ms
64 bytes from 172.16.200.1: icmp_seq=2 ttl=63 time=1.90 ms
64 bytes from 172.16.200.1: icmp_seq=3 ttl=63 time=2.02 ms
64 bytes from 172.16.200.1: icmp_seq=4 ttl=63 time=1.91 ms
64 bytes from 172.16.200.1: icmp_seq=5 ttl=63 time=2.24 ms
64 bytes from 172.16.200.1: icmp_seq=6 ttl=63 time=2.34 ms
64 bytes from 172.16.200.1: icmp_seq=7 ttl=63 time=2.14 ms

--- 172.16.200.1 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6010ms
rtt min/avg/max/mdev = 1.681/2.030/2.335/0.208 ms
```

Packet captures(PE1):

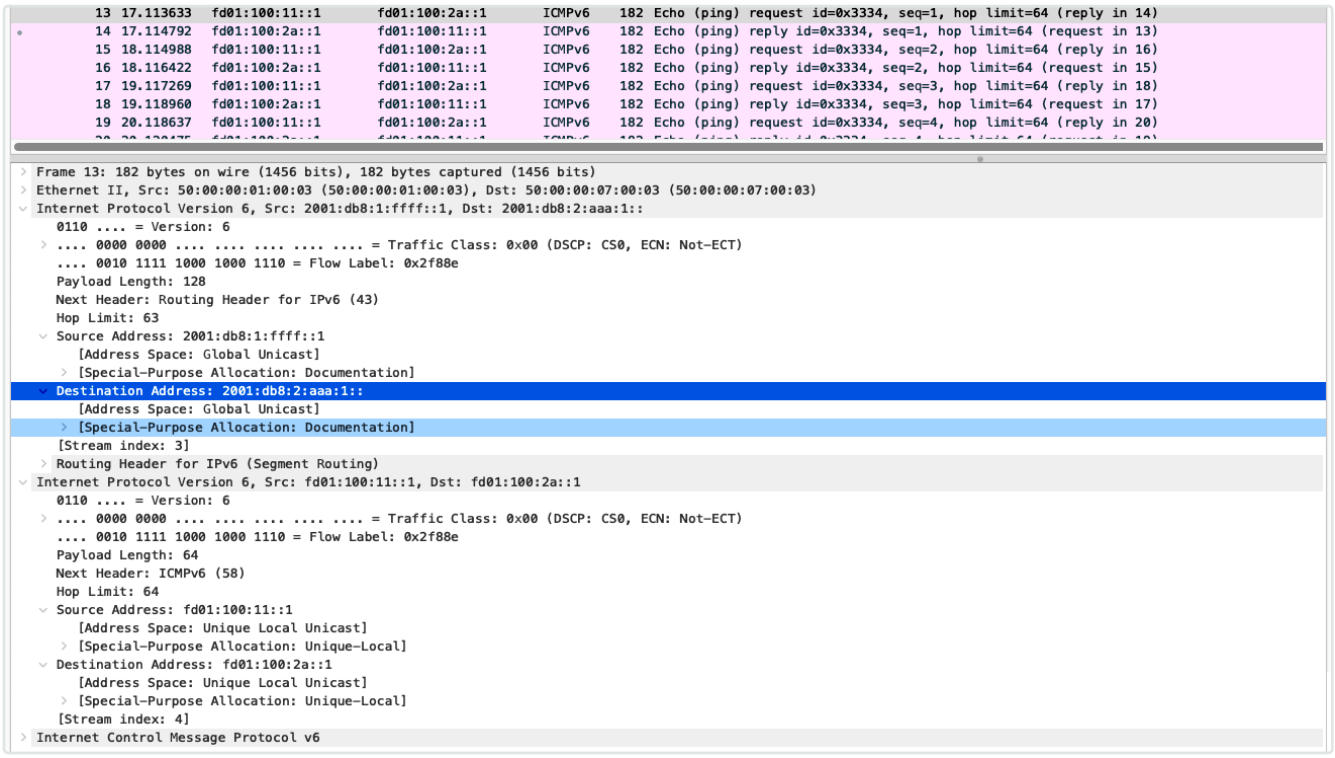


The ICMP echo request traffic from source **172.16.100.1** to destination **172.16.200.1** is encapsulated within an IPv6 packet using Segment Routing over IPv6 (SRv6). The outer IPv6 header specifies the destination address as **2001:db8:2:aaa:1::**, which corresponds to the **End.DT46** behavior. This SID indicates decapsulation and forwarding into **VRF customer-1**, where the original IPv4 packet is then routed to its destination.

IPv6:

```
vyos@CE1:~$ ping fd01:100:2a::1
PING fd01:100:2a::1(fd01:100:2a::1) 56 data bytes
64 bytes from fd01:100:2a::1: icmp_seq=1 ttl=63 time=1.53 ms
64 bytes from fd01:100:2a::1: icmp_seq=2 ttl=63 time=1.82 ms
64 bytes from fd01:100:2a::1: icmp_seq=3 ttl=63 time=2.34 ms
64 bytes from fd01:100:2a::1: icmp_seq=4 ttl=63 time=2.40 ms
64 bytes from fd01:100:2a::1: icmp_seq=5 ttl=63 time=2.09 ms
64 bytes from fd01:100:2a::1: icmp_seq=6 ttl=63 time=2.39 ms
^C
--- fd01:100:2a::1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 1.532/2.094/2.404/0.325 ms
vyos@CE1:~$
```

Packet capture:



Similar to the IPv4 test above, this capture shows ICMPv6 traffic encapsulated within an outer IPv6 header. The destination address **2001:db8:2:aaa:1::** corresponds to an **End.DT46** SID, indicating that the packet is decapsulated and forwarded into **VRF customer-1**. The inner payload remains ICMPv6, and the encapsulation is observable via **tcpdump** or Wireshark as SRv6 segment routing.