Configure Segment Routing over IPv6 (SRv6)

Table 1: Feature History Table

Feature Name	Release	Description
SRv6 with Micro-Segment (uSID)	Release 7.4.2	This release introduces support for Segment Routing over IPv6 data plane using Micro SIDs (uSIDs). This feature allows the source router to encode multiple SRv6 uSID instructions within a single 128-bit SID address. Such functionality allows for efficient and compact SRv6 SID representation with a low MTU overhead. The feature introduces the following commands: • segment-routing srv6 locators locator • segment-routing srv6 logging locator status • segment-routing srv6 sid holdtime • show segment-routing srv6 locator • show segment-routing srv6 manager

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

Segment Routing over IPv6 Overview

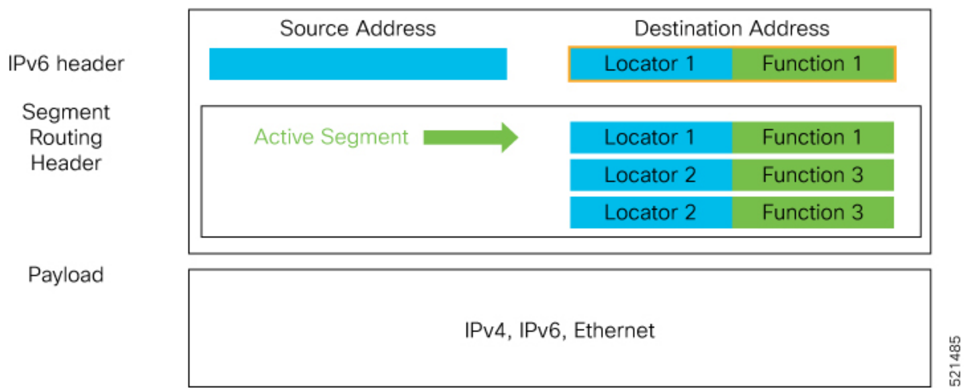
Segment Routing (SR) can be applied on both MPLS and IPv6 data planes. Segment Routing over IPv6 (SRv6) extends Segment Routing support with IPv6 data plane.

In an SR-MPLS enabled network, an MPLS label represents an instruction. The source nodes programs the path to a destination in the packet header as a stack of labels.

SRv6 introduces the Network Programming framework that enables a network operator or an application to specify a packet processing program by encoding a sequence of instructions in the IPv6 packet header. Each instruction is implemented on one or several nodes in the network and identified by an SRv6 Segment Identifier (SID) in the packet. The SRv6 Network Programming framework is defined in IETF RFC 8986 SRv6 Network Programming.

In SRv6, an IPv6 address represents an instruction. SRv6 uses a new type of IPv6 Routing Extension Header, called the Segment Routing Header (SRH), in order to encode an ordered list of instructions. The active segment is indicated by the destination address of the packet, and the next segment is indicated by a pointer in the SRH.

Figure 1: Network Program in the Packet Header



The SRv6 SRH is documented in IETF RFC IPv6 Segment Routing Header (SRH).

The SRH is defined as follows:

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  Last Entry   |     Flags     |              Tag              |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 |            Segment List[0] (128-bit IPv6 address)             |
 |                                                               |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 |                                                               |
                               ...
 |                                                               |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 |            Segment List[n] (128-bit IPv6 address)             |
 |                                                               |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 //                                                             //
 //         Optional Type Length Value objects (variable)       //
 //                                                             //
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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 *n*th 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.

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.

SRv6 Node Roles

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.
- Endpoint 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

SRv6 Head-End Behaviors

The SR Headend with Encapsulation behaviors are documented in the IETF RFC 8986 SRv6 Network Programming.

The SR Headend with Insertion head-end behaviors are documented in the following IETF draft:

https://datatracker.ietf.org/doc/draft-filsfils-spring-srv6-net-pgm-insertion/

This section describes a set of SR Policy headend behaviors. The following list summarizes them:

- H.Encaps—SR Headend Behavior with Encapsulation in an SRv6 Policy
- H.Encaps.Red—H.Encaps with Reduced Encapsulation
- H.Insert—SR Headend with insertion of an SRv6 Policy
- H.Insert.Red—H.Insert with reduced insertion

SRv6 Endpoint Behaviors

The SRv6 endpoint behaviors are documented in the IETF RFC 8986 SRv6 Network Programming.

The following is a subset of defined SRv6 endpoint behaviors that can be associated with a SID.

- End—Endpoint function. The SRv6 instantiation of a Prefix SID [RFC8402].
- End.X—Endpoint with Layer-3 cross-connect. The SRv6 instantiation of an Adj SID [RFC8402].
- End.DX6—Endpoint with decapsulation and IPv6 cross-connect (IPv6-L3VPN - equivalent to per-CE VPN label).
- End.DX4—Endpoint with decapsulation and IPv4 cross-connect (IPv4-L3VPN - equivalent to per-CE VPN label).
- End.DT6—Endpoint with decapsulation and IPv6 table lookup (IPv6-L3VPN - equivalent to per-VRF VPN label).
- End.DT4—Endpoint with decapsulation and IPv4 table lookup (IPv4-L3VPN - equivalent to per-VRF VPN label).
- End.DX2—Endpoint with decapsulation and L2 cross-connect (L2VPN use-case).
- End.B6.Encaps—Endpoint bound to an SRv6 policy with encapsulation. SRv6 instantiation of a Binding SID.
- End.B6.Encaps.RED—End.B6.Encaps with reduced SRH. SRv6 instantiation of a Binding SID.

SRv6 Endpoint Behavior Variants

Depending on how the SRH is handled, different behavior variants are defined for the End and End.X behaviors. The End and End.X behaviors can support these variants, either individually or in combinations.

- **Penultimate Segment Pop (PSP) of the SRH variant**—An SR Segment Endpoint Nodes receive the IPv6 packet with the Destination Address field of the IPv6 Header equal to its SID address.

  A penultimate SR Segment Endpoint Node is one that, as part of the SID processing, copies the last SID from the SRH into the IPv6 Destination Address and decrements the Segments Left value from one to zero.

The PSP operation takes place only at a penultimate SR Segment Endpoint Node and does not happen at non-penultimate endpoint nodes. When a SID of PSP-flavor is processed at a non-penultimate SR Segment Endpoint Node, the PSP behavior is not performed since Segments Left would not be zero.

The SR Segment Endpoint Nodes advertise the SIDs instantiated on them via control plane protocols. A PSP-flavored SID is used by the Source SR Node when it needs to instruct the penultimate SR Segment Endpoint Node listed in the SRH to remove the SRH from the IPv6 header.

- **Ultimate Segment Pop (USP) of the SRH variant**—The SRH processing of the End and End.X behaviors are modified as follows:

  If Segments Left is 0, then:

  1. Update the Next Header field in the preceding header to the Next Header value of the SRH
  2. Decrease the IPv6 header Payload Length by 8*(Hdr Ext Len+1)
  3. Remove the SRH from the IPv6 extension header chain
  4. Proceed to process the next header in the packet

  One of the applications of the USP flavor is when a packet with an SRH is destined to an application on hosts with smartNICs implementing SRv6. The USP flavor is used to remove the consumed SRH from the extension header chain before sending the packet to the host.

- **Ultimate Segment Decapsulation (USD) variant**—The Upper-layer header processing of the End, End.X and End.T behaviors are modified as follows:

  - **End** behavior: If the Upper-layer Header type is 41 (IPv6), then:

    1. Remove the outer IPv6 Header with all its extension headers
    2. Submit the packet to the egress IPv6 FIB lookup and transmission to the new destination
    3. Else, if the Upper-layer Header type is 4 (IPv4)
    4. Remove the outer IPv6 Header with all its extension headers
    5. Submit the packet to the egress IPv4 FIB lookup and transmission to the new destination
    6. Else, process as per Section 4.1.1 (Upper-Layer Header) of IETF RFC 8986 SRv6 Network Programming

  - **End.X** behavior: If the Upper-layer Header type is 41 (IPv6) or 4 (IPv4), then:

    1. Remove the outer IPv6 Header with all its extension headers
    2. Forward the exposed IP packet to the L3 adjacency J
    3. Else, process as per Section 4.1.1 (Upper-Layer Header) of IETF RFC 8986 SRv6 Network Programming

  One of the applications of the USD flavor is the case of TI-LFA in P routers with encapsulation with H.Encaps. The USD flavor allows the last Segment Endpoint Node in the repair path list to decapsulate the IPv6 header added at the TI-LFA Point of Local Repair and forward the inner packet.

SRv6 Micro-Segment (uSID)

The SRv6 micro-segment (uSID) is an extension of the SRv6 architecture. It leverages the SRv6 Network Programming architecture to encode several SRv6 Micro-SID (uSID) instructions within a single 128-bit SID address. Such a SID address is called a uSID Carrier.

SRv6 uSID is documented in the IETF drafts Network Programming extension: SRv6 uSID instruction and Compressed SRv6 Segment List Encoding in SRH.

Throughout this chapter, we will refer to SRv6 micro-segment as "uSID".

The SRv6 uSID provides the following benefits:

- Leverages the SRv6 Network Programming with no change. SRv6 uSID is a new pseudo code in the existing SRv6 network programming framework.
- Leverages the SRv6 data plane (SRH) with no change. Any SID in the destination address or SRH can be an SRv6 uSID carrier.
- Leverages the SRv6 control plane with no change.
- Ultra-Scale—Scalable number of globally unique nodes in the domain, for example:
  - 16-bit uSID ID size: 65k uSIDs per domain block
  - 32-bit uSID ID size: 4.3M uSIDs per domain block
- Lowest MTU overhead
  - 6 uSIDs per uSID carrier
  - For example, 18 source-routing waypoints in only 40 bytes of overhead
- Hardware-friendliness:
  - Leverages mature hardware capabilities (inline IP Destination Address edit, IP Destination Address longest match).
  - Avoids any extra lookup in indexed mapping tables.
  - A micro-program with 6 or fewer uSIDs requires only legacy IP-in-IP encapsulation behavior.
- Scalable Control Plane:
  - Summarization at area/domain boundary provides massive scaling advantage.
  - No routing extension is required, a simple prefix advertisement suffices.
- Seamless Deployment:
  - A uSID may be used as a SID (the carrier holds a single uSID).
  - The inner structure of an SR Policy can stay opaque to the source. A carrier with uSIDs is just seen as a SID by the policy headend Security.
  - Leverages SRv6's native SR domain security.

SRv6 uSID Terminology

The SRv6 Network Programming is extended with the following terms:

- uSID—An identifier that specifies a micro-segment.

  A uSID has an associated behavior that is the SRv6 function (for example, a node SID or Adjacency SID) associated with the given ID. The node at which an uSID is instantiated is called the "Parent" node.

- uSID Carrier—A 128-bit IPv6 address (carried in either in the packet destination address or in the SRH) in the following format:

  ```
  <uSID-Block><Active-uSID><Next-uSID>...<Last-uSID><End-of-Carrier>...<End-of-Carrier>
  ```

  where:

  - uSID Block—An IPv6 prefix that defines a block of SRv6 uSIDs.
  - Active uSID—The first uSID that follows the uSID block.
  - Next uSID—The next uSID after the Active uSID.
  - Last uSID—The last uSID in the carrier before the End-of-Carrier uSID.
  - End-of-Carrier —A globally reserved uSID that marks the end of a uSID carrier. The End-of-Carrier ID is **0000**. All empty uSID carrier positions must be filled with the End-of-Carrier ID; therefore, a uSID carrier can have more than one End-of-Carrier.

The following is an example of an SRH with 3 Micro-SID carriers for a total of up to 18 micro-instructions:

Micro-SID Carrier1: {uInstruction1, uInstruction2… uInstruction6}
Micro-SID Carrier2: {uInstruction7, uInstruction8… uInstruction12}
Micro-SID Carrier3: {uInstruction13, uInstruction14… uInstruction18}

SRv6 uSID Carrier Format

The uSID carrier format specifies the type of uSID carrier supported in an SRv6 network. The format specification includes Block size and ID size.

- **uSID Block**

  The uSID block is an IPv6 prefix that defines a block of SRv6 uSIDs. This can be an IPv6 prefix allocated to the provider (for example, /22, /24, and so on.), or it can be any well-known IPv6 address block generally available for private use, such as the ULA space FC/8, as defined in IETF draft RFC4193.

  An SRv6 network may support more than a single uSID block.

  The length of block [prefix] is defined in bits. From a hardware-friendliness perspective, it is expected to use sizes on byte boundaries (16, 24, 32, and so on).

- **uSID ID**

  The length of uSID ID is defined in bits. From a hardware-friendliness perspective, it is expected to use sizes on byte boundaries (8, 16, 24, 32, and so on).

The uSID carrier format is specified using the notation "F*bbuu*" , where “*bb*” is size of block and “*uu*” is size of ID. For example, "F3216" is a format with a 32-bit uSID block and 16-bit uSID IDs.

SRv6 uSID Allocation Within a uSID Block

The architecture for uSID specifies both globally scoped and locally scoped uSIDs, where a globally scoped uSID is the type of uSID that provides reachability to the node.

On the other hand, a locally scoped uSID is associated to a local behavior, and therefore *must* be preceded by a globally scoped uSID of the parent node when relying on routing to forward the packet.

The Global ID block (GIB) is the set of IDs available for globally scoped uSID allocation. The Local ID block (LIB) is the set of IDs available for locally scoped uSID allocation.

A globally scoped uSID is a uSID from the GIB. A globally scoped uSID typically identifies a shortest path to a node in the SR domain. An IP route (for example, /48) is advertised by the parent node to each of its globally scoped uSIDs, under the associated uSID block. The parent node executes a variant of the END behavior.

The “Nodal” uSID (uN) is an example of a globally scoped behavior defined in uSID architecture.

A node can have multiple globally scoped uSIDs under the same uSID blocks (for example, one per IGP flex-algorithm). Multiple nodes may share the same globally scoped uSID (Anycast).

A locally scoped uSID is a uSID from the LIB. A locally scoped uSID identifies a local micro-instruction on the parent node; for example, it may identify a cross-connect to a direct neighbor over a specific interface or a VPN context. Locally scoped uSIDs are not routeable.

For example, if N1 and N2 are two different physical nodes of the uSID domain and *L* is a locally scoped uSID value, then N1 and N2 may bind two different behaviors to *L*.

The uSIDs are allocated in one of following ways: auto, dynamic, or explicit.

- The request to allocate locally scoped uSIDs comes from SRv6 clients (such as IS-IS or BGP). The request can be to allocate any available ID (dynamic allocation) or to allocate a specific ID (explicit allocation).

SRv6 Endpoint Behaviors Associated with uSID

The SRv6 Network Programming is extended with new types of SRv6 SID endpoint behaviors:

- uN—A short notation for the NEXT-CSID (Compressed SID) End behavior with a pseudocode of shift-and-lookup
- uA—A short notation for the NEXT-CSID End.X behavior with a pseudocode of shift-and-xconnect
- uDT—A short notation for the NEXT-CSID End.DT behavior with the same pseudocode as End.DT4/End.DT6/End.DT2U/End.DT2M
- uDX—A short notation for the NEXT-CSID End.DX behavior with the same pseudocode as End.DX4/End.DX6/End.DX2

SRv6 uSID in Action - Example

This example highlights an integrated VPN and Traffic Engineering use-case leveraging SRv6 uSID.

VPNv4 site A connected to Node 1 sends packets to VPNv4 site B connected to Node 2 alongside a traffic engineered path via Node 8 and Node 7 using a single 128-bit SRv6 SID.

Node 1 is the ingress PE; Node 2 is the egress PE.

Nodes 3, 4, 5, and 6 are classic IPv6 nodes. Traffic received on these nodes use classic IP forwarding without changing the outer DA.

Nodes 1, 8, 7 and 2 are SRv6 capable configured with:

- 32-bit SRv6 block = fcbb:bb01
- 16-bit SRv6 ID

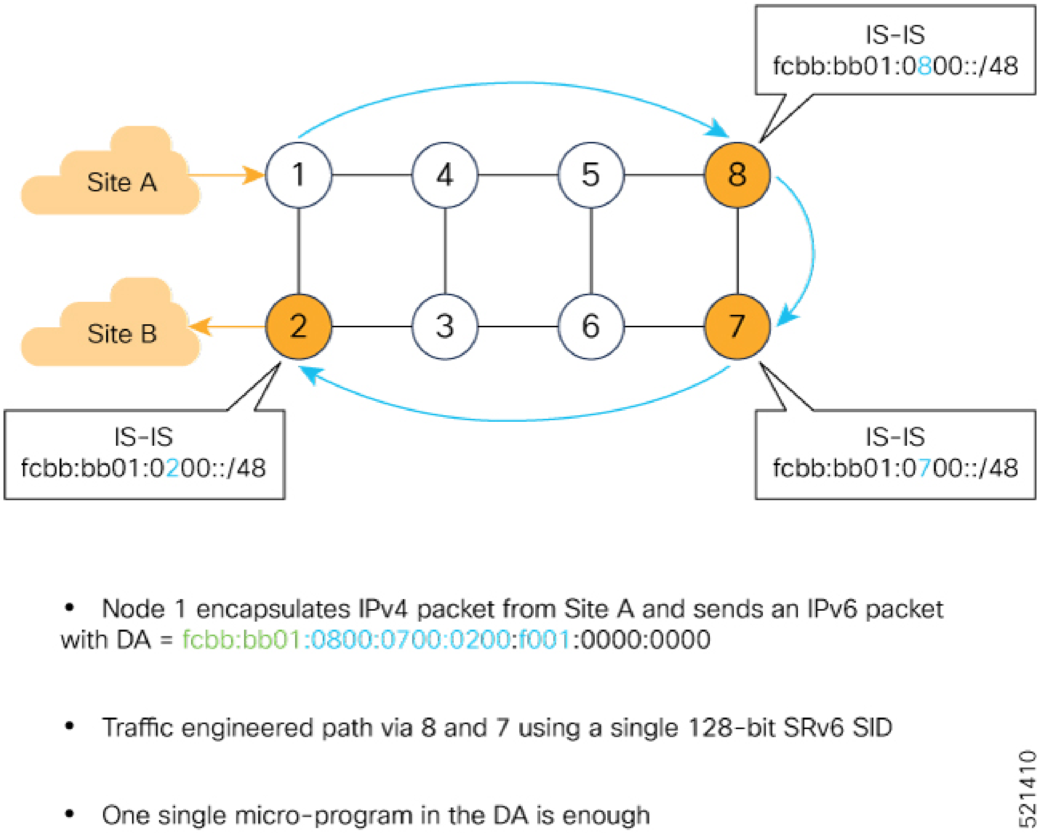
  For example:

  - Node 7 uN = fcbb:bb01:0700::/48
  - Node 8 uN = fcbb:bb01:0800::/48

The following IGP routes are advertised:

- Node 8 advertises the IGP route fcbb:bb01:**0800**::/48
- Node 7 advertises the IGP route fcbb:bb01:**0700**::/48
- Node 2 advertises the IGP route fcbb:bb01:**0200**::/48

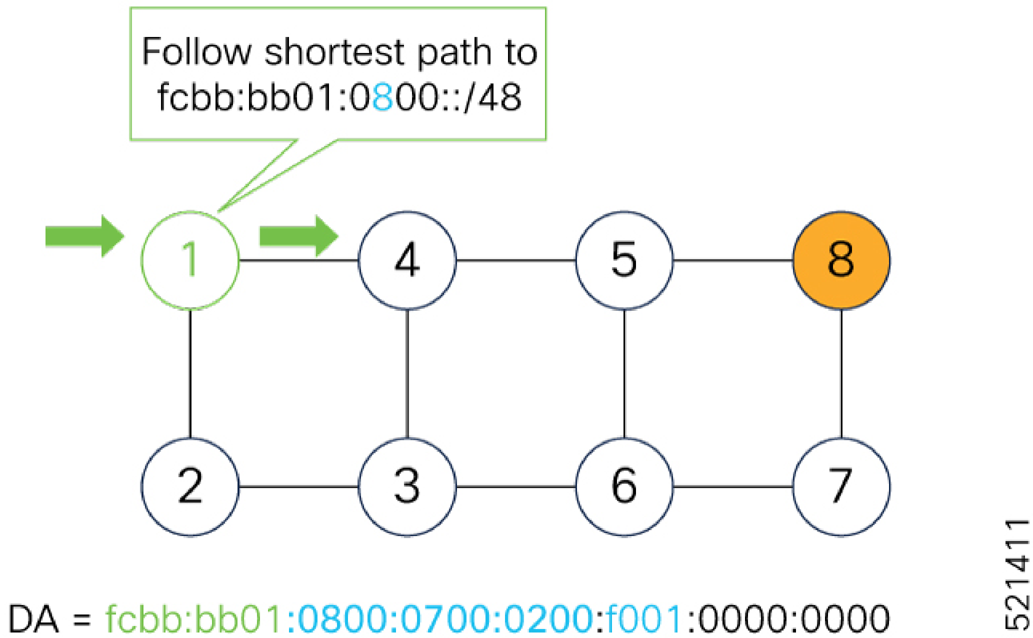
Figure 2: Integrated VPN and Traffic Engineering SRv6 uSID Use-case



Node 1 encapsulates an IPv4 packet from VPN Site A and sends an IPv6 packet with destination address fcbb:bb01:**0800**:**0700**:**0200**:f001:0000:0000. This is a uSID carrier, with a list of micro-instructions (uSIDs) (0800, 0700, 0200, f001, and 0000 – indicating the end of the instruction).

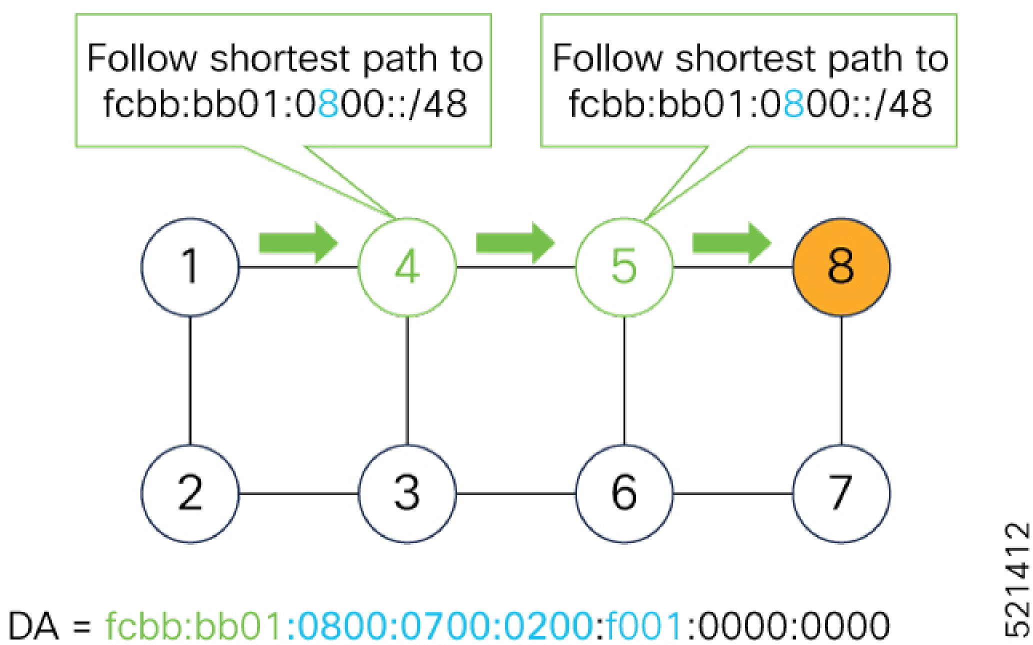
uSIDs (uNs) 0800, 0700, 0200 are used to realize the traffic engineering path to Node 2 with way points at Nodes 8 and 7. uSID f001 is the BGP-signalled instruction (uDT4) advertized by Node 2 for the VPNv4 service

Figure 3: Node 1: End.B6.Encaps Behavior



Nodes 4 and 5 simply forward the packet along the shortest path to Node 8, providing seamless deployment through classic IPv6 nodes.

Figure 4: Node 4 and Node 5: Classic IPv6 Nodes

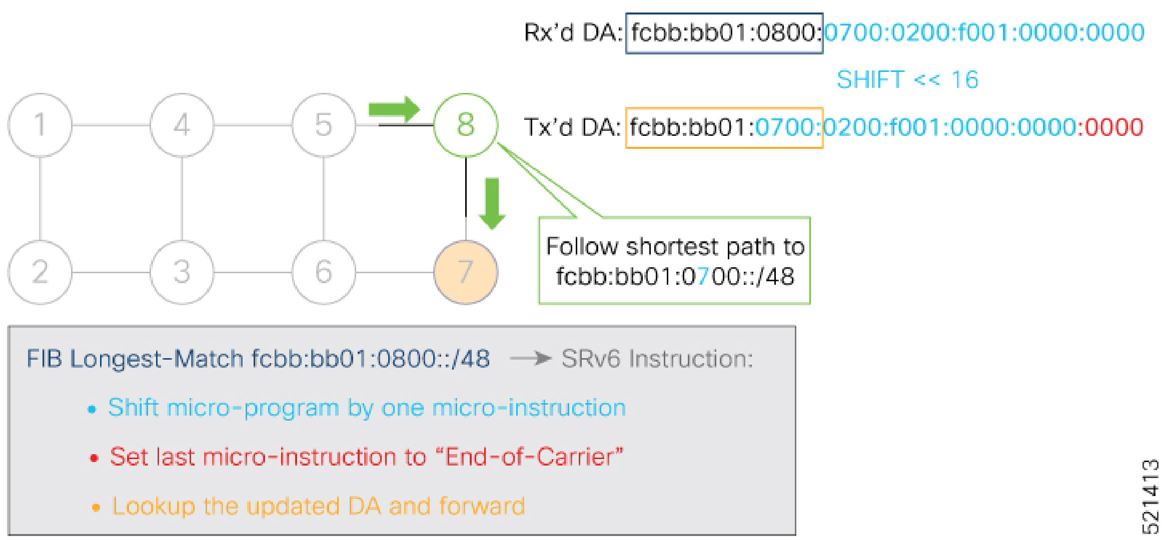


When Node 8 receives the packet, it performs SRv6 uN behavior (shift-and-lookup with PSP/USD). It removes its outer DA (0800) and advances the micro program to the next micro instruction by doing the following:

1. Pops its own uSID (0800)

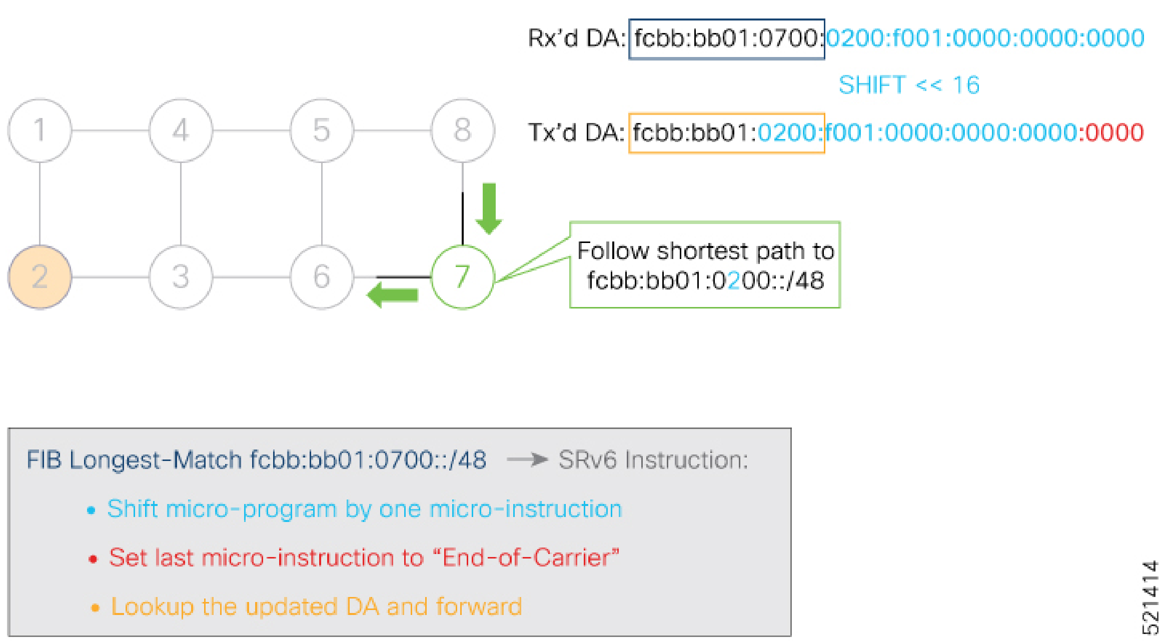
2. **Shifts** the remaining DA by 16-bits to the left
3. Fills the remaining bits with 0000 (End-of-Carrier)
4. Performs a **lookup** for the shortest path to the next DA (fcbb:bb01:**0700::/48**)
5. Forwards it using the new DA fcbb:bb01:**0700**:0200:f001:0000:0000:0000

Figure 5: Node 8: SRv6 uN Behavior (Shift and Forward)



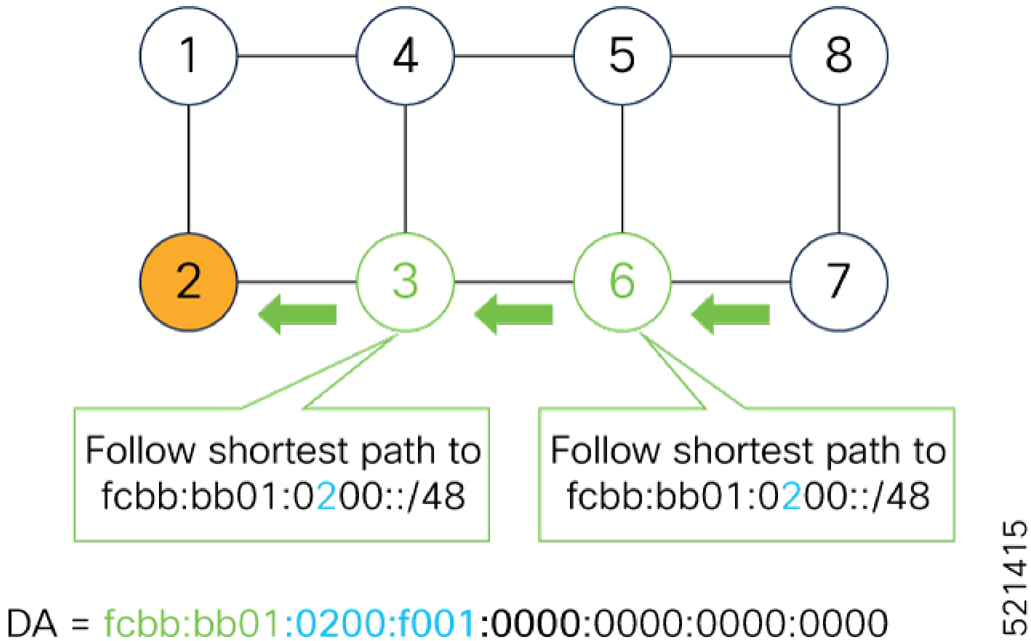
When Node 7 receives the packet, it performs the same SRv6 uN behavior (shift-and-lookup with PSP/USD), forwarding it using the new DA fcbb:bb01:**0200**:f001:0000:0000:0000:0000

Figure 6: Node 7: SRv6 uN Behavior (Shift and Forward)



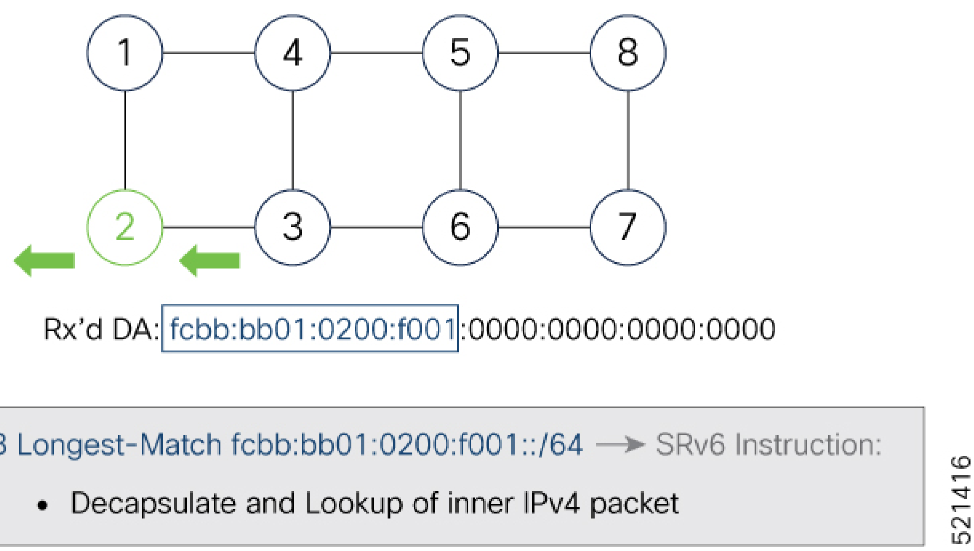
Nodes 6 and 3 simply forward the packet along the shortest path to Node 2, providing seamless deployment through classic IPv6 nodes.

Figure 7: Node 6 and Node 3: Classic IPv6 Nodes



When Node 2 receives the packet, it performs an SRv6 uDT4 behavior (End.DT4—Endpoint with decapsulation and IPv4 table lookup) to VPNv4 Site B.

Figure 8: Node 2: SRv6 uDT4 Behavior



To recap, this example showed an integrated VPN and Traffic Engineering use-case, where VPNv4 site A connected to Node 1 sent packets to VPNv4 site B connected to Node 2 alongside a traffic engineered path via Node 8 and Node 7 using a single 128-bit SRv6 SID:

- @1: inner packet P encapsulated with outer DA fcbb:bb01:**0800**:**0700**:**0200**:f001:0000:0000

- @4 & @5: classic IP forwarding, outer DA unchanged
- @8: SRv6 uN behavior: shift and lookup, outer DA becomes fcbb:bb01:**0700**:**0200**:f001:0000:0000:0000
- @7: SRv6 uN behavior: shift and lookup, outer DA becomes fcbb:bb01:**0200**:f001:0000:0000:0000:0000
- @6 & @3: classic IP forwarding, outer DA unchanged
- @2: SRv6 End.DT4: Decapsulate and IPv4 table lookup

Usage Guidelines and Limitations

General Guidelines and Limitations

- Cisco IOS XR supports uSIDs with 32-bit uSID block and 16-bit uSID IDs (3216).

  A single UCF format must be used for uSID locators in a SRv6 uSID domain.
- Cisco IOS XR supports up to 8 uSID locator prefixes.
- Cisco IOS XR supports uSID locator prefixes from different uSID blocks.

  Up to 64 uSID blocks can be used across all uSID locators in the network.
- Cisco IOS XR Release 7.4.2 supports the following SRv6 uSID behaviors:
  - uN (shift)
  - uA (shift)
- SRv6 over GRE interface is not supported
- SRv6 over BVI interface is not supported

uSID Allocation Recommendation

We recommend that the uSID block allocation is made from the IPv6 Unique Local Address (ULA) range.

Note Allocation from the public Global Unicast Addresses (GUA) range is also supported.

- Use ULA /24 base from FC00::/8 space
  - FC*BB*:*BB*/24, with *B* indicating a nibble value picked by operator
- 256 uSID blocks possible from this allocation
  - In this release, 64 uSID blocks are supported
  - FC*BB*:*BBVV*/32, with *VV* two variable nibbles. The supported values for *VV* in Cisco IOS XR Release 7.3.1 are 0x00 to 0x3F.

For example:

- ULA /24 base = FC00:01/24
- uSID block space = 64 uSID blocks (from FC00:01**00**/32 to FC00:01**3F**/32)

Platform-Specific Guidelines and Limitations

This release supports SRv6 on NCS 6000 series routers.

- The NCS 6000 series router can act as transit router on the SRv6 uSID network, performing uN/uA (shift/lookup) operation.
- The following line cards are supported:
  - NC6-60x10GE-L-S, NC6-60x10GE-M-S
  - NC6-10X100G-L-K, NC6-10X100G-L-P, NC6-10X100G-M-P, NC6-10X100G-M-K
  - NC6-20X100GE-L-C, NC6-20X100GE-M-C
- The NCS 6000 series router cannot act as service PE
- PSP, USP behaviors for uN and uA are not supported
- TI-LFA is not supported
- Microloop Avoidance is not supported

Configuring SRv6

Enabling SRv6 involves the following high-level configuration steps:

- Configure SRv6 locator(s)
- Enable SRv6 under IS-IS

Configure SRv6 Locator Name, Prefix, and uSID-Related Parameters

This section shows how to globally enable SRv6 and configure locator.

- **segment-routing srv6 locators locator** *locator*—Globally enable SRv6 and configure the locator.
- **segment-routing srv6 locators locator** *locator* **prefix** *ipv6_prefix*/*length*—Configure the locator prefix value.

The following example shows how to globally enable SRv6 and configure a locator.

```
Router(config)# segment-routing srv6
Router(config-srv6)# locators
Router(config-srv6-locators)# locator myLoc1
Router(config-srv6-locator)# prefix 2001:0:8::/48
```

(Optional) Customize SRv6 Logging for Locator Status Changes

- **segment-routing srv6 logging locator status**—Enable the logging of locator status.

(Optional) Customize SRv6 SID Parameters

- **segment-routing srv6 sid holdtime** *minutes*—The holdtime for a stale or freed SID. The range of *minutes* is from 0 (disabled) to 60 minutes.

The following example shows how to configure optional SRv6 parameters:

```
RP/0/RSP0/CPU0:Node1(config-srv6)# logging locator status
RP/0/RSP0/CPU0:Node1(config-srv6)# sid holdtime 10

RP/0/RSP0/CPU0:Node1(config-srv6)#
```

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.

```
Router# show segment-routing srv6 manager

Parameters:
  SRv6 Enabled: Yes
  SRv6 Operational Mode:
    Micro-segment:
      SID Base Block: 2001::/24
  Encapsulation:
    Source Address:
      Configured: ::
      Default: 1:1:1::1
    Hop-Limit: Default
    Traffic-class: Default
Summary:
  Number of Locators: 5 (5 operational)
  Number of SIDs: 12 (0 stale)
  Max SIDs: 9000
  OOR:
    Thresholds: Green 450, Warning 270
    Status: Resource Available
        History: (0 cleared, 0 warnings, 0 full)
    Block 2001::/32:
        Number of SIDs free: 7674
        Max SIDs: 7680
        Thresholds: Green 384, Warning 231
        Status: Resource Available
            History: (0 cleared, 0 warnings, 0 full)
Platform Capabilities:
  SRv6: Yes
  TILFA: No
  Microloop-Avoidance: No
  Endpoint behaviors:
    uN (shift)
    uA (shift)
  Headend behaviors:
    None
  Security rules:
    None
  Counters:
    None
  Signaled parameters:
    Max-SL          : 0
    Max-End-Pop-SRH : 0
    Max-H-Insert    : 0 sids
    Max-H-Encap     : 0 sids
```

```
    Max-End-D       : 0
  Configurable parameters (under srv6):
    Encapsulation:
      Source Address: No
      Hop-Limit     : value=No, propagate=No
      Traffic-class : value=No, propagate=No
  Max SIDs: 9000
  SID Holdtime: 3 mins
```

Verifying SRv6 Locator

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

```
Router# show segment-routing srv6 locator myLoc1 detail

Name                  ID       Algo  Prefix                    Status   Flags
--------------------  -------  ----  ------------------------  -------  --------
myLoc1                2        0     2001:0:8::/48             Up       U
  (U): Micro-segment (behavior: uN (shift))
  Interface:
    Name: srv6-myLoc1
    IFH : 0x0800002c
    IPv6 address: 2001:0:8::/48
  Number of SIDs: 1
  Created: Jan 11 14:22:30.141 (2w5d ago)
```

Verifying SRv6 SIDs

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

```
Router# show segment-routing srv6 locator myLoc1 sid
SID                   Behavior     Context                    Owner           State  RW
--------------------  -----------  ------------------------   --------------  -----  --
2001:0:8::            uN (shift)   'default':1                sidmgr          InUse  Y
```

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

```
Router# show segment-routing srv6 locator myLoc1 sid 2001:0:8:: detail
SID                   Behavior     Context                    Owner           State  RW
--------------------  -----------  ------------------------   --------------  -----  --
2001:0:8::            uN (shift)   'default':1                sidmgr          InUse  Y
  SID Function: 0x1
  SID context: { table-id=0xe0800000 ('default':IPv6/Unicast), opaque-id=1 }
  Locator: 'myLoc1'
  Allocation type: Dynamic
  Created: Jan 11 14:22:30.490 (2w5d ago)
```

Similarly, you can display SID information across locators by using the **show segment-routing srv6 sid** command.

Configuring SRv6 under 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 IS-IS 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.

Usage Guidelines and Restrictions

The following usage guidelines and restrictions apply for SRv6 IS-IS:

- An IS-IS address-family can support either SR-MPLS or SRv6, but both at the same time is not supported.

Configuring SRv6 under IS-IS

To configure SRv6 IS-IS, use the following command:

- **router isis** *instance* **address-family ipv6 unicast segment-routing srv6 locator** *locator* [**level** {**1** | **2**}]—Enable SRv6 under the IS-IS IPv6 address-family and assign SRv6 locator(s) to it. Use the **level** {**1** | **2**} keywords to advertise the locator only in the specified IS-IS level.

The following example shows how to configure SRv6 under IS-IS.

```
Router(config)# router isis core
Router(config-isis)# address-family ipv6 unicast
Router(config-isis-af)# segment-routing srv6
Router(config-isis-srv6)# locator myLoc1 level 1
Router(config-isis-srv6-loc)# exit
```

For more information about configuring IS-IS, refer to the "Implemeting IS-IS" chapter in the *Routing Configuration Guide for Cisco NCS 6000 Series Routers*.