SPRING
Internet-Draft
Intended status: Standards Track
Expires: March 22, 2020

C. Filsfils, Ed.
P. Camarillo, Ed.
Cisco Systems, Inc.
J. Leddy
Individual Contributor
D. Voyer
Bell Canada
S. Matsushima
SoftBank
Z. Li
Huawei Technologies
September 19, 2019

# SRv6 Network Programming
## draft-ietf-spring-srv6-network-programming-02

## Abstract

This document describes the SRv6 network programming concept and its most basic functions.

## Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 22, 2020.

Table of Contents

## 1. Introduction

Segment Routing leverages the source routing paradigm. An ingress node steers a packet through an ordered list of instructions, called segments. Each one of these instructions represents a function to be called at a specific location in the network. A function is locally defined on the node where it is executed and may range from simply moving forward in the segment list to any complex user-defined behavior. The network programming consists in combining segment routing functions, both simple and complex, to achieve a networking objective that goes beyond mere packet routing.

This document defines the SRv6 Network Programming concept and aims at standardizing the main segment routing behaviors to enable the creation of interoperable overlays with underlay optimization and service programming.

The companion document [I-D.filsfils-spring-srv6-net-pgm-illustration] illustrates the concepts defined in this document.

Familiarity with the Segment Routing Header [I-D.ietf-6man-segment-routing-header] is assumed.

## 2. Terminology

Terminology used within this document is defined in detail in [RFC8402]. Specifically, the terms: Segment Routing, SR Domain, SRv6, Segment ID (SID), SRv6 SID, Active Segment, and SR Policy.

SRH: Segment Routing Header as defined in [I-D.ietf-6man-segment-routing-header]. We assume that the SRH may be present multiple times inside each packet.

NH: Next-header field of the IPv6 header. NH=SRH means that the next-header of the IPv6 header is Routing Header for IPv6(43) with the Type field set to 4.

SL: The Segments Left field of the SRH

FIB: Forwarding Information Base. A FIB lookup is a lookup in the forwarding table.

SA: Source Address

DA: Destination Address





An SR Policy is resolved to a SID list. A SID list is represented as <S1, S2, S3> where S1 is the first SID to visit, S2 is the second SID to visit and S3 is the last SID to visit along the SR path.

(SA,DA) (S3, S2, S1; SL) represents an IPv6 packet with:

- Source Address is SA, Destination Address is DA, and next-header is SRH

- SRH with SID list <S1, S2, S3> with Segments Left = SL

- Note the difference between the <> and () symbols: <S1, S2, S3> represents a SID list where S1 is the first SID and S3 is the last SID to traverse. (S3, S2, S1; SL) represents the same SID list but encoded in the SRH format where the rightmost SID in the SRH is the first SID and the leftmost SID in the SRH is the last SID. When referring to an SR policy in a high-level use-case, it is simpler to use the <S1, S2, S3> notation. When referring to an illustration of the detailed packet behavior, the (S3, S2, S1; SL) notation is more convenient.

- The payload of the packet is omitted.

When a packet is intercepted on a wire, it is possible that SRH[SL] is different from the DA.

## 3. SRv6 SID

As introduced in RFC8402 an SRv6 Segment Identifier is a 128-bit value.

When an SRv6 SID is in the Destination Address field of an IPv6 header of a packet, it is routed through an IPv6 network as an IPv6 address.

Its processing is defined in [I-D.ietf-6man-segment-routing-header] section 4.3 and reproduced here as a reminder.

> Without constraining the details of an implementation, the SR segment endpoint node creates Forwarding Information Base (FIB) entries for its local SIDs.
>
> When an SRv6-capable node receives an IPv6 packet, it performs a longest-prefix-match lookup on the packets destination address. This lookup can return any of the following:
>
> - A FIB entry that represents a locally instantiated SRv6 SID





> - A FIB entry that represents a local interface, not locally instantiated as an SRv6 SID
>
> - A FIB entry that represents a non-local route
>
> - No Match

This document formally defines behaviors and parameters for SRv6 SIDs.

### 3.1. SID format

An SRv6 SID is represented as LOC:FUNCT where LOC (locator) is the L most significant bits and FUNCT (function) is the 128-L least significant bits of the SID. L is called the locator length and is

flexible. Each operator is free to use the locator length it chooses. Most often the locator is routable and leads to the node which instantiates that SID. A control-plane protocol might represent the locator as B:N where B is the SRv6 SID block (IPv6 subnet allocated for SRv6 SIDs by the operator) and N is the identifier of the parent node.

The function part of the SID is an opaque identification of a local behavior bound to the SID. The FUNCT value zero is invalid.

The terminology "function" refers to the bit-string in the SRv6 SID. The terminology "behavior" identifies the pseudocode bound to the SID. The behaviors are defined in Section 4 of this document.

A behavior may require additional arguments that would be placed immediately after the FUNCT. In such case, the SRv6 SID will have the form LOC:FUNCT:ARGS::. For this reason, the SRv6 SIDs are matched on a per longest-prefix-match basis.

ARG may vary on a per-packet basis and may contain information related to the flow, service, or any other information required by FUNCT. The ARG value of a routed SID SHOULD remain constant among packets in a given flow. Varying ARG values among packets in a flow may result in different ECMP hashing and cause re-ordering.

### 3.2. SID reachability

Most often, the node N would advertise IPv6 prefix(es) matching the LOC parts covering its SIDs or shorter-mask prefix. The distribution of these advertisements and calculation of their reachability are routing protocol specific aspects that are outside the scope of this document.





An SRv6 SID is said to be routed if its SID belongs to an IPv6 prefix advertised via a routing protocol. An SRv6 SID that does not fulfill this condition is non-routed.

Let's provide a classic illustration:

Node N is configured explictly with two SIDs: 2001:DB8:B:1:100:: and 2001:DB8:B:2:101::.

The network learns about a path to 2001:DB8:B:1::/64 via the IGP and hence a packet destined to 2001:DB8:B:1:100:: would be routed up to N. The network does not learn about a path to 2001:DB8:B:2::/64 via the IGP and hence a packet destined to 2001:DB8:B:2:101:: would not be routed up to N.

A packet could be steered to a non-routed SID 2001:DB8:B:2:101:: by using a SID list <...,2001:DB8:B:1:100::,2001:DB8:B:2:101::,...> where the non-routed SID is preceded by a routed SID to the same node. Routed and non-routed SRv6 SIDs are the SRv6 instantiation of global and local segments, respectively [RFC8402].





## 4. Behaviors associated with a SID

Each FIB entry indicates the behavior associated with the a SID instance and its parameters.

We define hereafter a set of well-known behaviors that can be associated with a SID.

| | |
|---|---|
| End | Endpoint function<br>The SRv6 instantiation of a prefix SID |
| End.X | Endpoint with Layer-3 cross-connect<br>The SRv6 instantiation of a Adj SID |
| End.T | Endpoint with specific IPv6 table lookup |
| End.DX6 | Endpoint with decaps and IPv6 cross-connect<br>e.g. IPv6-L3VPN (equivalent to per-CE VPN label) |
| End.DX4 | Endpoint with decaps and IPv4 cross-connect<br>e.g. IPv4-L3VPN (equivalent to per-CE VPN label) |
| End.DT6 | Endpoint with decaps and IPv6 table lookup<br>e.g. IPv6-L3VPN (equivalent to per-VRF VPN label) |
| End.DT4 | Endpoint with decaps and IPv4 table lookup<br>e.g. IPv4-L3VPN (equivalent to per-VRF VPN label) |
| End.DT46 | Endpoint with decaps and IP table lookup<br>e.g. IP-L3VPN (equivalent to per-VRF VPN label) |
| End.DX2 | Endpoint with decaps and L2 cross-connect<br>e.g. L2VPN use-case |
| End.DX2V | Endpoint with decaps and VLAN L2 table lookup<br>e.g. EVPN Flexible cross-connect use-case |
| End.DT2U | Endpoint with decaps and unicast MAC L2table lookup<br>e.g. EVPN Bridging unicast use-case |
| End.DT2M | Endpoint with decaps and L2 table flooding<br>e.g. EVPN Bridging BUM use-case with ESI filtering |
| End.B6.Insert | Endpoint bound to an SRv6 policy<br>SRv6 instantiation of a Binding SID |
| End.B6.Insert.RED | [...] with reduced SRH insertion<br>SRv6 instantiation of a Binding SID |
| End.B6.Encaps | Endpoint bound to an SRv6 policy with encaps<br>SRv6 instantiation of a Binding SID |
| End.B6.Encaps.RED | [...] with reduced SRH insertion<br>SRv6 instantiation of a Binding SID |
| End.BM | Endpoint bound to an SR-MPLS Policy<br>SRv6 instantiation of an SR-MPLS Binding SID |

The list is not exhaustive. In practice, any function can be attached to a local SID: e.g. a node N can bind a SID to a local VM or container which can apply any complex processing on the packet.

The following subsections detail the behavior that a node (N) binds to a SID (S).

At the end of this section, we also present some flavors of these well-known behaviors.

## 4.1. End: Endpoint

The Endpoint behavior ("End" for short) is the most basic behavior. It is the instantiation of a Prefix-SID [RFC8402].

It does not allow for decapsulation of an outer header nor the removal of an SRH. As a consequence, an End SID cannot be the last SID of a SID list and cannot be the DA of a packet without an SRH (unless combined with the PSP, USP or USD flavors Section 4.18).

The following defines SRH processing and, if SRH is not present, upper-layer header processing when a matched FIB entry represents a locally instantiated End SID.

When N receives a packet whose IPv6 DA is S and S is a local End SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Send an ICMP Parameter Problem message to the Source Address
             Code TBD-SRH (SR Upper-layer Header Error),
             Pointer set to the offset of the upper-layer header,
             interrupt packet processing and discard the packet
S04.   }
S05.   If (IPv6 Hop Limit <= 1) {
S06.      Send an ICMP Time Exceeded message to the Source Address,
             Code 0 (Hop limit exceeded in transit),
             interrupt packet processing and discard the packet
S07.   }
S08.   max_LE = (Hdr Ext Len / 2) - 1
S09.   If ((Last Entry > max_LE) or (Segments Left > Last Entry+1)) {
S10.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S11.   }
S12.   Decrement Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Update IPv6 DA with Segment List[Segments Left]
S15.   Resubmit the packet to the egress IPv6 FIB lookup and
          transmission to the new destination
S16. }
```

Notes:





The End behavior operates on the same FIB table (i.e. VRF, L3 relay id) associated to the packet. Hence the FIB lookup on line S15 is done in the same FIB table as the ingress interface.

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End SID, send an ICMP parameter problem message to the Source Address and discard the packet. Error code TBD-SRH (SR Upper-layer Header Error) and Pointer set to the offset of the upper-layer header.

## 4.2. End.X: Layer-3 cross-connect

The "Endpoint with cross-connect to an array of layer-3 adjacencies" behavior (End.X for short) is a variant of the End behavior.

It is the SRv6 instantiation of an Adjacency-SID [RFC8402] and it is required to express any traffic-engineering policy.

An instance of the End.X behavior is associated with a set of J of one or more Layer-3 adjacencies.

When N receives a packet destined to S and S is a local End.X SID, the line S15 from the End processing is replaced by the following:

S15. Set the packet's egress adjacency to J

Notes:
S15. If the set J contains several L3 adjacencies, then one element of the set is selected based on the hash of the packet's header Section 6.2.

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.X SID, send an ICMP parameter problem message to the Source Address and discard the packet. Error code "SR Upper-layer Header Error", Pointer set to the offset of the upper-layer header.

Note that the End.X SID cannot be the last SID of a SID list and cannot be the DA of a packet without an SRH (unless combined with the PSP, USP or USD flavors Section 4.18). Hence the Upper-layer header should never be processed.

If a node N has 30 outgoing interfaces to 30 neighbors, usually the operator would explicitly instantiate 30 End.X SIDs at N: one per





layer-3 adjacency to a neighbor. Potentially, more End.X could be explicitly defined (groups of layer-3 adjacencies to the same neighbor or to different neighbors).

Note that if N has an outgoing interface bundle I to a neighbor Q made of 10 member links, N may allocate up to 11 End.X local SIDs for that bundle(LAG): one for the bundle(LAG) itself and then up to one for each member link.

## 4.3. End.T: Specific IPv6 table lookup

The "Endpoint with specific IPv6 table lookup" behavior (End.T for short) is a variant of the End behavior.

The End.T behavior is used for multi-table operation in the core. For this reason, an instance of the End.T behavior must be associated with an IPv6 FIB table T.

When N receives a packet destined to S and S is a local End.T SID, the line S15 from the End processing is replaced by the following:

S15.1. Set the packet's associated FIB table to T
S15.2. Resubmit the packet to the egress IPv6 FIB lookup and transmission to the new destination

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.T SID, send an ICMP parameter problem message to the Source Address and discard the packet. Error code "SR Upper-layer Header Error", Pointer set to the offset of the upper-layer header.

Note that the End.T SID cannot be the last SID of a SID list and cannot be the DA of a packet without an SRH (unless combined with the PSP, USP or USD flavors Section 4.18). Hence the Upper-layer header should never be processed.





### 4.4. End.DX6: Decapsulation and IPv6 cross-connect

The "Endpoint with decapsulation and cross-connect to an array of IPv6 adjacencies" behavior (End.DX6 for short) is a variant of the End.X behavior.

One of the applications of the End.DX6 behavior is the L3VPNv6 use-case where a FIB lookup in a specific tenant table at the egress PE is not required. This is equivalent to the per-CE VPN label in MPLS [RFC4364].

The End.DX6 SID must be the last segment in a SR Policy, and it must be associated with one or more L3 IPv6 adjacencies J.

When N receives a packet destined to S and S is a local End.DX6 SID, N does the following processing:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S04.   }
```

```
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.DX6 SID, the following must be done.

```
S01. If (Upper-Layer Header type != 41) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the outer IPv6 Header with all its extension headers
S05. Forward the exposed IPv6 packet to the L3 adjacency J
```

Notes:
S01. 41 refers to IPv6 encapsulation as defined by IANA allocation for Internet Protocol Numbers.
S05.  If the End.DX6 SID is bound to an array of L3 adjacencies, then one entry of the array is selected based on the hash of the packet's header Section 6.2.





### 4.5. End.DX4: Decapsulation and IPv4 cross-connect

The "Endpoint with decapsulation and cross-connect to an array of IPv4 adjacencies" behavior (End.DX4 for short) is a variant of the End.X behavior.

One of the applications of the End.DX4 behavior is the L3VPNv4 use-case where a FIB lookup in a specific tenant table at the egress PE is not required.  This is equivalent to the per-CE VPN label in MPLS [RFC4364].

The End.DX4 SID must be the last segment in a SR Policy, and it must be associated with one or more L3 IPv4 adjacencies J.

When N receives a packet destined to S and S is a local End.DX4 SID, N does the following processing:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.DX4 SID, the following must be done.

```
S01. If (Upper-Layer Header type != 4) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the outer IPv6 Header with all its extension headers
S05. Forward the exposed IPv4 packet to the L3 adjacency J
```

Notes:
S01. 4 refers to IPv4 encapsulation as defined by IANA allocation for Internet Protocol Numbers
S05. If the End.DX4 SID is bound to an array of L3 adjacencies, then one entry of the array is selected based on the hash of the packet's header Section 6.2.





### 4.6. End.DT6: Decapsulation and specific IPv6 table lookup

The "Endpoint with decapsulation and specific IPv6 table lookup" behavior (End.DT6 for short) is a variant of the End.T behavior.

One of the applications of the End.DT6 behavior is the L3VPNv6 use-case where a FIB lookup in a specific tenant table at the egress PE is required. This would be equivalent to the per-VRF VPN label in MPLS [RFC4364].

Note that an End.DT6 may be defined for the main IPv6 table in which

case and End.DT6 supports the equivalent of an IPv6inIPv6 decapsulation (without VPN/tenant implication).

The End.DT6 SID must be the last segment in a SR Policy, and a SID instance must be associated with an IPv6 FIB table T.

When N receives a packet destined to S and S is a local End.DT6 SID, N does the following processing:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.DT6 SID, N does the following:

```
S01. If (Upper-Layer Header type != 41) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the outer IPv6 Header with all its extension headers
S05. Set the packet's associated FIB table to T
S06. Resubmit the packet to the egress IPv6 FIB lookup and
        transmission to the new destination
```





### 4.7. End.DT4: Decapsulation and specific IPv4 table lookup

The "Endpoint with decapsulation and specific IPv4 table lookup" behavior (End.DT4 for short) is a variant of the End behavior.

One of the applications of the End.DT4 behavior is the L3VPNv4 use-case where a FIB lookup in a specific tenant table at the egress PE is required. This would be equivalent to the per-VRF VPN label in MPLS [RFC4364].

Note that an End.DT4 may be defined for the main IPv4 table in which case an End.DT4 supports the equivalent of an IPv4inIPv6 decapsulation (without VPN/tenant implication).

The End.DT4 SID must be the last segment in a SR Policy, and a SID instance must be associated with an IPv4 FIB table T.

When N receives a packet destined to S and S is a local End.DT4 SID, N does the following processing:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.DT4 SID, N does the following:

```
S01. If (Upper-Layer Header type != 4) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the outer IPv6 Header with all its extension headers
S05. Set the packet's associated FIB table to T
S06. Resubmit the packet to the egress IPv4 FIB lookup and
        transmission to the new destination
```

### 4.8. End.DT46: Decapsulation and specific IP table lookup

The "Endpoint with decapsulation and specific IP table lookup" behavior (End.DT46 for short) is a variant of the End.DT4 and End.DT6 behavior.

One of the applications of the End.DT46 behavior is the L3VPN use-case where a FIB lookup in a specific IP tenant table at the egress PE is required. This would be equivalent to single per-VRF VPN label (for IPv4 and IPv6) in MPLS[RFC4364].

Note that an End.DT46 may be defined for the main IP table in which case an End.DT46 supports the equivalent of an IPinIPv6 decapsulation(without VPN/tenant implication).

The End.DT46 SID must be the last segment in a SR Policy, and a SID instance must be associated with an IPv4 FIB table T4 and an IPv6 FIB table T6.

When N receives a packet destined to S and S is a local End.DT46 SID, N does the following processing:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.DT46 SID, N does the following:

```
S01. If (Upper-layer Header type == 4) {
S02.    Remove the outer IPv6 Header with all its extension headers
S03.    Set the packet's associated FIB table to T4
S04.    Resubmit the packet to the egress IPv4 FIB lookup and
           transmission to the new destination
S05. } Else if (Upper-layer Header type == 41) {
S06.    Remove the outer IPv6 Header with all its extension headers
S07.    Set the packet's associated FIB table to T6
S08.    Resubmit the packet to the egress IPv6 FIB lookup and
           transmission to the new destination
S09. } Else {
S10.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S11. }
```

### 4.9. End.DX2: Decapsulation and L2 cross-connect

The "Endpoint with decapsulation and Layer-2 cross-connect to an outgoing L2 interface (OIF)" (End.DX2 for short) is a variant of the endpoint behavior.

One of the applications of the End.DX2 behavior is the L2VPN/EVPN VPWS use-case.

The End.DX2 SID must be the last segment in a SR Policy, and it must be associated with one outgoing interface J.

When N receives a packet destined to S and S is a local End.DX2 SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S04.   }
```

```
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.DX2 SID, the following must be done.





```
S01. If (Upper-Layer Header type != 59) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the outer IPv6 Header with all its extension headers and
        forward the ethernet frame to the OIF J.
```

Notes:
S01.  The next-header value 59 identifies that there is no further Internet Protocol header to be processed in the packet.  When the SID corresponds to the End.DX2 and the Next-Header value is 59, we know that an Ethernet frame is directly in the payload without any further header.
S04.  If the SID S is bound to an array of L2 OIFs then one entry of the array is selected based on a hash of the packet's header Section 6.2.
S04.  An End.DX2 behavior could be customized to expect a specific VLAN format and rewrite the egress VLAN header before forwarding on the outgoing interface.

### 4.10. End.DX2V: Decapsulation and VLAN L2 table lookup

The "Endpoint with decapsulation and specific VLAN table lookup" behavior (End.DX2V for short) is a variant of the End.DX2 behavior.

One of the applications of the End.DX2V behavior is the EVPN Flexible cross-connect use-case.  The End.DX2V behavior is used to perform a lookup of the ethernet frame VLANs in a particular L2 table.  Any SID instance of the End.DX2V behavior must be associated with an L2 Table T.

When N receives a packet whose IPv6 DA is S and S is a local End.DX2 SID, the processing is identical to the End.DX2 behavior except for the Upper-layer header processing which is modified as follows:

```
S04. Remove the outer IPv6 Header with all its extension headers,
        lookup the exposed inner VLANs in L2 table T, and forward
        via the matched table entry.
```

Notes:
An End.DX2V behavior could be customized to expect a specific VLAN format and rewrite the egress VLAN header before forwarding on the outgoing interface.





### 4.11. End.DT2U: Decapsulation and unicast MAC L2 table lookup

The "Endpoint with decapsulation and specific unicast MAC L2 table lookup" behavior (End.DT2U for short) is a variant of the End behavior.

One of the applications of the End.DT2U behavior is the EVPN Bridging unicast . Any SID instance of the End.DT2U behavior must be associated with an L2 Table T.

When N receives a packet whose IPv6 DA is S and S is a local End.DT2U SID, the processing is identical to the End.DX2 behavior except for the Upper-layer header processing which is as follows:

```
S01. If (Upper-Layer Header type != 59) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the IPv6 header and all its extension headers
S05. Learn the exposed inner MAC Source Address in L2 Table T
S06. Lookup the exposed inner MAC Destination Address in L2 Table T
S07. If (matched entry in T) {
S08.    Forward via the matched table T entry
```

```
S09. } Else {
S10.    Forward via all L2OIFs entries in table T
S11. }
```

Notes:
S05. In EVPN, the learning of the exposed inner MAC SA is done via control plane.

## 4.12. End.DT2M: Decapsulation and L2 table flooding

The "Endpoint with decapsulation and specific L2 table flooding" behavior (End.DT2M for short) is a variant of the End.DT2U behavior.

One of the applications of the End.DT2M behavior is the EVPN Bridging BUM with ESI filtering use-case.

Any SID instance of this behavior must be associated with a L2 table T. Additionally the behavior may take an argument: "Arg.FE2". It is an argument specific to EVPN ESI filtering used to exclude a specific OIF (or set of OIFs) from L2 table T flooding.





When N receives a packet whose IPv6 DA is S and S is a local End.DT2M SID, the processing is identical to the End.DT2M behavior except for the Upper-layer header processing which is as follows:

```
S01. If (Upper-Layer Header type != 59) {
S02.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S03. }
S04. Remove the IPv6 header and all its extension headers
S05. Learn the exposed inner MAC Source Address in L2 Table T
S06. Forward via all L2 OIFs excluding the one specified in Arg.F2
```

Notes:
S05. In EVPN, the learning of the exposed inner MAC SA is done via control plane

## 4.13. End.B6.Insert: Endpoint bound to an SRv6 policy

The "Endpoint bound to an SRv6 Policy" is a variant of the End behavior.

One of its applications is to express scalable traffic-engineering policies across multiple domains. It is the one of the SRv6 instantiations of a Binding SID [RFC8402].

An End.B6.Insert SID is never the last segment in a SID list, and any SID instantiation must be associated with an SR Policy B[I-D.ietf-spring-segment-routing-policy].

When N receives a packet whose IPv6 DA is S and S is a local End.B6.Insert SID, does:





```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Send an ICMP Parameter Problem message to the Source Address
             Code TBD-SRH (SR Upper-layer Header Error),
             Pointer set to the offset of the upper-layer header,
             interrupt packet processing and discard the packet
S04.   }
S04.   If (IPv6 Hop Limit <= 1) {
S05.       Send an ICMP Time Exceeded message to the Source Address,
             Code 0 (Hop limit exceeded in transit),
             interrupt packet processing and discard the packet
S06.   }
```

```
S07.   max_LE = (Hdr Ext Len / 2) - 1
S08.   If ((Last Entry > max_LE) or (Segments Left > (Last Entry+1)){
S09.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S11.   }
S12.   Decrement Hop Limit by 1
S13.   Insert a new SRH in between the IPv6 Header and the received
        SRH containing the list of segments of B
S14.   Set the IPv6 DA to the first segment of B
S15.   Resubmit the packet to the egress IPv6 FIB lookup and
          transmission to the new destination
S16. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.B6.Insert SID, send an ICMP parameter problem message to the Source Address and discard the packet. Error code "SR Upper-layer Header Error", Pointer set to the offset of the upper-layer header.

### 4.14. End.B6.Insert.Red: [...] with reduced SRH

This is an optimization of the End.B6.Insert behavior.

End.B6.Insert.Red reduces the size of the new SRH by one SID by avoiding the insertion of the first SID in the pushed SRH. In this way, the first SID is only written in the DA and the packet is forwarded according to it.

The new SRH is created as described in Section 4.1.1 of [I-D.ietf-6man-segment-routing-header].





### 4.15. End.B6.Encaps: Endpoint bound to an SRv6 policy w/ encaps

This is a variation of the End behavior.

One of its applications is to express scalable traffic-engineering

policies across multiple domains. It is the one of the SRv6 instantiations of a Binding SID [RFC8402].

Instead of simply inserting an SRH with the policy (End.B6.Insert), this behavior also adds an outer IPv6 header.

An End.B6.Encaps SID is never the last segment in a SID list. Any SID instantiation must be associated with an SR Policy B[I-D.ietf-spring-segment-routing-policy] and a source address A.

When N receives a packet whose IPv6 DA is S and S is a local End.B6.Encaps SID, does:

```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Send an ICMP Parameter Problem message to the Source Address
             Code TBD-SRH (SR Upper-layer Header Error),
             Pointer set to the offset of the upper-layer header,
             interrupt packet processing and discard the packet
S04.   }
S05.   If (IPv6 Hop Limit <= 1) {
S06.       Send an ICMP Time Exceeded message to the Source Address,
             Code 0 (Hop limit exceeded in transit),
             interrupt packet processing and discard the packet
S07.   }
S08.   max_LE = (Hdr Ext Len / 2) - 1
S09.   If ((Last Entry > max_LE) or (Segments Left > (Last Entry+1)) {
S10.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S11.   }
S12.   Decrement Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Push a new IPv6 header with its own SRH containing B
S15.   Set the outer IPv6 SA to A
S16.   Set the outer IPv6 DA to the first SID of B
S17.   Set the outer PayloadLength, Traffic Class, FlowLabel and
          Next-Header fields
S18.   Resubmit the packet to the egress IPv6 FIB lookup and
          transmission to the new destination
S19. }
```

Notes:
S13. The SRH MAY be omitted when the SRv6 Policy B only contains one SID and there is no need to use any flag, tag or TLV.
S16. The Payload Length, Traffic Class and Next-Header fields are set as per [RFC2473]. The Flow Label is computed as per [RFC6437].

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.B6.Encaps SID, send an ICMP parameter problem message to the Source Address and discard the packet. Error code "SR Upper-layer Header Error", Pointer set to the offset of the upper-layer header.

### 4.16. End.B6.Encaps.Red: [...] with reduced SRH

This is an optimization of the End.B6.Encaps behavior.

End.B6.Encaps.Red reduces the size of the SRH by one SID by avoiding the insertion of the first SID in the outer SRH. In this way, the first segment is only introduced in the DA and the packet is forwarded according to it.

The new SRH is created as described in Section 4.1.1 of [I-D.ietf-6man-segment-routing-header].

The SRH MAY be omitted when the SRv6 Policy only contains one segment and there is no need to use any flag, tag or TLV.

### 4.17. End.BM: Endpoint bound to an SR-MPLS policy

The "Endpoint bound to an SR-MPLS Policy" is a variant of the End behavior.

The End.BM behavior is required to express scalable traffic-engineering policies across multiple domains where some domains support the MPLS instantiation of Segment Routing. This is an SRv6 instantiation of an SR-MPLS Binding SID [RFC8402].

An End.BM SID is never the last SID, and any SID instantiation must be associated with an SR-MPLS Policy B[I-D.ietf-spring-segment-routing-policy].

When N receives a packet whose IPv6 DA is S and S is a local End.BM SID, does:

```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Send an ICMP Parameter Problem message to the Source Address
             Code TBD-SRH (SR Upper-layer Header Error),
             Pointer set to the offset of the upper-layer header,
             interrupt packet processing and discard the packet
S04.   }
S05.   If (IPv6 Hop Limit <= 1) {
S06.      Send an ICMP Time Exceeded message to the Source Address,
             Code 0 (Hop limit exceeded in transit),
             interrupt packet processing and discard the packet
S07.   }
S08.   max_LE = (Hdr Ext Len / 2) - 1
S09.   If ((Last Entry > max_LE) or (Segments Left > (Last Entry+1)) {
S10.      Send an ICMP Parameter Problem to the Source Address,
             Code 0 (Erroneous header field encountered),
             Pointer set to the Segments Left field,
             interrupt packet processing and discard the packet
S11.   }
S12.   Decrement Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Push a the MPLS label stack for B
S15.   Submit the packet to the MPLS engine for transmission to the
          topmost label.
S16. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 End.BM SID, send an ICMP parameter problem message to the Source Address and discard the packet.  Error code "SR Upper-layer Header Error", Pointer set to the offset of the upper-layer header.

### 4.18. Flavors

The PSP, USP and USD flavors are variants of the End, End.X and End.T behaviors. For each of these behaviors these flavors may be supported for a SID either individually or in combinations.

#### 4.18.1. PSP: Penultimate Segment Pop of the SRH

The SRH processing of the End, End.X and End.T behaviors are modified: after the instruction "S14. Update IPv6 DA with Segment List[Segments Left]" is executed, the following instructions must be executed as well:

```
S14.1.   If (updated SL == 0) {
S14.2.      Pop the SRH
S14.3.   }
```

#### 4.18.2. USP: Ultimate Segment Pop of the SRH

The SRH processing of the End, End.X and End.T behaviors are modified: the instructions S02-S04 are substituted by the following ones:

```
S02.   If (Segments Left == 0) {
S03.       Pop the SRH
S04.   }
```

#### 4.18.3. USD: Ultimate Segment Decapsulation

The SRH processing of the End, End.X and End.T behaviors are modified: the instructions S02-S04 are substituted by the following ones:

```
S02.   If (Segments Left == 0) {
S03.      Skip the SRH processing and proceed to the next header
```

```
S04.   }
```

Further on, the Upper-layer header processing of the End, End.X and End.T behaviors are modified as follows:





```
End:
S01. If (Upper-layer Header type == 41) {
S02.    Remove the outer IPv6 Header with all its extension headers
S03.    Resubmit the packet to the egress IPv6 FIB lookup and
           transmission to the new destination
S04. } Else {
S05.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S06. }

End.T:
S01. If (Upper-layer Header type == 41) {
S02.    Remove the outer IPv6 Header with all its extension headers
S03.    Set the packet's associated FIB table to T
S04.    Resubmit the packet to the egress IPv6 FIB lookup and
           Transmission to the new destination
S05. } Else {
S06.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S07. }

End.X:
S01. If (Upper-layer Header type == 41) {
S02.    Remove the outer IPv6 Header with all its extension headers
```

```
S03.    Forward the exposed IPv6 packet to the L3 adjacency J
S04. } Else {
S05.    Send an ICMP Parameter Problem message to the Source Address
           Code TBD-SRH (SR Upper-layer Header Error),
           Pointer set to the offset of the upper-layer header,
           interrupt packet processing and discard the packet
S06. }
```





## 5. Transit behaviors

We define hereafter the set of basic transit behaviors. These behaviors are not bound to a SID and they correspond to source SR nodes or transit nodes [I-D.ietf-6man-segment-routing-header].

| | |
|---|---|
| T | Transit behavior |
| T.Insert | Transit behavior with insertion of an SRv6 policy |
| T.Insert.Red | Transit behavior with reduced insert of an SRv6 policy |
| T.Encaps | Transit behavior with encapsulation in an SRv6 policy |
| T.Encaps.Red | Transit behavior with reduced encaps in an SRv6 policy |
| T.Encaps.L2 | T.Encaps applied to received L2 frames |
| T.Encaps.L2.Red | T.Encaps.Red applied to received L2 frames |

This list can be expanded in case any new functionality requires it.

### 5.1. T: Transit behavior

As per [RFC8200], if a node N receives a packet (A, S2)(S3, S2, S1; SL=1) and S2 is neither a local address nor a local SID of N then N forwards the packet without inspecting the SRH.

This means that N treats the following two packets with the same performance:

- (A, S2)

- (A, S2)(S3, S2, S1; SL=1)

A transit node does not need to count by default the amount of transit traffic with an SRH extension header. This accounting might be enabled as an optional behavior.

A transit node MUST include the outer flow label in its ECMP load-balancing hash [RFC6437].

## 5.2. T.Insert: Transit with insertion of an SRv6 Policy

Node N receives two packets P1=(A, B2) and P2=(A,B2)(B3, B2, B1; SL=1). B2 is neither a local address nor SID of N.

N steers the transit packets P1 and P2 into an SRv6 Policy with one SID list <S1, S2, S3>.

The "T.Insert" transit insertion behavior is defined as follows:





1. insert the SRH (B2, S3, S2, S1; SL=3) ;; Ref1, Ref1bis
2. set the IPv6 DA = S1
3. forward along the shortest path to S1

Ref1: The received IPv6 DA is placed as last SID of the inserted SRH.

Ref1bis: The SRH is inserted [I-D.voyer-6man-extension-header-insertion] before any other IPv6 Routing Extension Header.

After the T.Insert behavior, P1 and P2 respectively look like:

-(A, S1) (B2, S3, S2, S1; SL=3)

-(A, S1) (B2, S3, S2, S1; SL=3) (B3, B2, B1; SL=1)

### 5.3. T.Insert.Red: Transit with reduced insertion

The T.Insert.Red behavior is an optimization of the T.Insert behavior. It is defined as follows:

1. insert the SRH (B2, S3, S2; SL=3)
2. set the IPv6 DA = S1
3. forward along the shortest path to S1

T.Insert.Red will reduce the size of the SRH by one segment by avoiding the insertion of the first SID in the pushed SRH. In this way, the first segment is only introduced in the DA and the packet is forwarded according to it.

After the T.Insert.Red behavior, P1 and P2 respectively look like:

- (A, S1) (B2, S3, S2; SL=3)

- (A, S1) (B2, S3, S2; SL=3) (B3, B2, B1; SL=1)

### 5.4. T.Encaps: Transit with encapsulation in an SRv6 Policy

Node N receives two packets P1=(A, B2) and P2=(A,B2)(B3, B2, B1; SL=1). B2 is neither a local address nor SID of N.

N steers the transit packets P1 and P2 into an SR Encapsulation Policy with a Source Address T and a Segment list <S1, S2, S3>.

The T.Encaps transit encapsulation behavior is defined as follows:





1. push an IPv6 header with its own SRH (S3, S2, S1; SL=2)
2. set outer IPv6 SA = T and outer IPv6 DA = S1
3. set outer payload length, traffic class and flow label ;; Ref1,2
4. update the Next-Header value ;; Ref1
5. decrement inner Hop Limit or TTL ;; Ref1
6. forward along the shortest path to S1

After the T.Encaps behavior, P1 and P2 respectively look like:

- (T, S1) (S3, S2, S1; SL=2) (A, B2)

- (T, S1) (S3, S2, S1; SL=2) (A, B2) (B3, B2, B1; SL=1)

The T.Encaps behavior is valid for any kind of Layer-3 traffic. This behavior is commonly used for L3VPN with IPv4 and IPv6 deployments.

The SRH MAY be omitted when the SRv6 Policy only contains one segment and there is no need to use any flag, tag or TLV.

Ref 1: As described in [RFC2473] (Generic Packet Tunneling in IPv6 Specification)

Ref 2: As described in [RFC6437] (IPv6 Flow Label Specification)

### 5.5. T.Encaps.Red: Transit with reduced encapsulation

The T.Encaps.Red behavior is an optimization of the T.Encaps behavior. It is defined as follows:

```
1.  push an IPv6 header with its own SRH (S3, S2; SL=2)
2.  set outer IPv6 SA = T and outer IPv6 DA = S1
3.  set outer payload length, traffic class and flow label   ;; Ref1,2
4.  update the Next-Header value                             ;; Ref1
5.  decrement inner Hop Limit or TTL                         ;; Ref1
6.  forward along the shortest path to S1
```

Ref 1: As described in [RFC2473] (Generic Packet Tunneling in IPv6 Specification)

Ref 2: As described in [RFC6437] (IPv6 Flow Label Specification)

T.Encaps.Red will reduce the size of the SRH by one segment by avoiding the insertion of the first SID in the SRH of the pushed IPv6 packet. In this way, the first segment is only introduced in the DA and the packet is forwarded according to it.

After the T.Encaps.Red behavior, P1 and P2 respectively look like:

- (T, S1) (S3, S2; SL=2) (A, B2)

- (T, S1) (S3, S2; SL=2) (A, B2) (B3, B2, B1; SL=1)

The SRH MAY be omitted when the SRv6 Policy only contains one segment and there is no need to use any flag, tag or TLV.

## 5.6. T.Encaps.L2: Transit with encapsulation of L2 frames

While T.Encaps encapsulates the received IP packet, T.Encaps.L2 encapsulates the received L2 frame (i.e. the received ethernet header and its optional VLAN header is in the payload of the outer packet).

If the outer header is pushed without SRH, then the DA must be a SID of type End.DX2, End.DX2V, End.DT2U or End.DT2M and the next-header must be 59 (IPv6 No Next Header [RFC8200]). The received Ethernet frame follows the IPv6 header and its extension headers.

Else, if the outer header is pushed with an SRH, then the last SID of the SRH must be of type End.DX2, End.DX2V, End.DT2U or End.DT2M and the next-header of the SRH must be 59 (IPv6 No Next Header [RFC8200]). The received Ethernet frame follows the IPv6 header and its extension headers.

The SRH MAY be omitted when the SRv6 Policy only contains one segment and there is no need to use any flag, tag or TLV.

## 5.7. T.Encaps.L2.Red: Transit with reduced encaps of L2 frames

The T.Encaps.L2.Red behavior is an optimization of the T.Encaps.L2 behavior.

T.Encaps.L2.Red will reduce the size of the SRH by one segment by avoiding the insertion of the first SID in the SRH of the pushed IPv6 packet. In this way, the first segment is only introduced in the DA and the packet is forwarded according to it.

The SRH MAY be omitted when the SRv6 Policy only contains one segment and there is no need to use any flag, tag or TLV.

# 6. Operation

## 6.1. Counters

Any SRv6 capable node SHOULD implement the following set of combined counters (packets and bytes):

- CNT-1: Per local SID entry, traffic that matched that SID and was processed correctly.

- CNT-2: Per SRv6 Policy, traffic steered into it and processed correctly.

Furthermore, an SRv6 capable node maintains an aggregate counter CNT-3 tracking the IPv6 traffic that was received with a destination address matching a local interface address that is not a locally instantiated SID and the next-header is SRH with SL>0. We remind that this traffic is dropped as an interface address is not a local SID by default. A SID must be explicitly instantiated.

## 6.2. Flow-based hash computation

When a flow-based selection within a set needs to be performed, the source address, the destination address and the flow-label MUST be included in the flow-based hash.

This occurs when the destination address is updated, a FIB lookup is performed and multiple ECMP paths exist to the updated destination address.

This occurs when End.X, End.DX4, or End.DX6 are bound to an array of adjacencies.

This occurs when the packet is steered in an SR policy whose selected path has multiple SID lists [I-D.ietf-spring-segment-routing-policy].

Additionally, any transit router in an SRv6 domain MUST include the outer flow label in its ECMP load-balancing hash [RFC6437].

## 6.3. OAM

[I-D.ali-spring-srv6-oam] defines the OAM behavior for SRv6. This includes the definition of the SRH Flag 'O-bit', as well as

additional OAM Endpoint behaviors.





## 7. Basic security for intra-domain deployment

We use the following terminology:

An internal node is a node part of the domain of trust.

A border router is an internal node at the edge of the domain of trust.

An external interface is an interface of a border router towards another domain.

An internal interface is an interface entirely within the domain of trust.

The internal address space is the IP address block dedicated to internal interfaces.

An internal SID is a SID instantiated on an internal node.

The internal SID space is the IP address block dedicated to internal SIDs.

External traffic is traffic received from an external interface to the domain of trust.

Internal traffic is traffic that originates and ends within the domain of trust.

The purpose of this section is to document how a domain of trust can operate SRv6-based services for internal traffic while preventing any external traffic from accessing the internal SRv6-based services.

It is expected that future documents will detail enhanced security mechanisms for SRv6 (e.g. how to allow external traffic to leverage internal SRv6 services).

### 7.1. SEC-1

An SRv6 router MUST support an ACL on the external interface that drops any traffic with SA or DA in the internal SID space.

A provider would generally do this for its internal address space to prevent access to internal addresses and in order to prevent spoofing. The technique is extended to the local SID space.

The typical counters of an ACL are expected.





### 7.2. SEC-2

An SRv6 router MUST support an ACL with the following behavior:

1. IF (DA == LocalSID) && (SA != internal address or SID space)
2. drop

This prevents access to locally instantiated SIDs from outside the operator's infrastructure. Note that this ACL may not be enabled in all cases. For example, specific SIDs can be used to provide resources to devices that are outside of the operator's infrastructure.

The typical counters of an ACL are expected.

### 7.3. SEC-3

As per the End definition, an SRv6 router MUST only implement the End behavior on a local IPv6 address if that address has been explicitly enabled as an SRv6 SID.

This address may or may not be associated with an interface. This address may or may not be routed. The only thing that matters is that the local SID must be explicitly instantiated and explicitly bound to a behavior.

Packets received with destination address representing a local interface that has not been enabled as an SRv6 SID MUST be dropped.

# 8. Control Plane

In an SDN environment, one expects the controller to explicitly provision the SIDs and/or discover them as part of a service discovery function. Applications residing on top of the controller could then discover the required SIDs and combine them to form a distributed network program.

The concept of "SRv6 network programming" refers to the capability for an application to encode any complex program as a set of individual functions distributed through the network. Some functions relate to underlay SLA, others to overlay/tenant, others to complex applications residing in VM and containers.

The specification of the SRv6 control-plane is outside the scope of this document.

We limit ourselves to a few important observations.





## 8.1. IGP

The End, End.T and End.X SIDs express topological behaviors and hence are expected to be signaled in the IGP together with the flavors PSP, USP and USD[I-D.bashandy-isis-srv6-extensions].

The presence of SIDs in the IGP do not imply any routing semantics to the addresses represented by these SIDs. The routing reachability to an IPv6 address is solely governed by the classic, non-SID-related, IGP information. Routing is not governed neither influenced in any way by a SID advertisement in the IGP.

These three SIDs provide important topological behaviors for the IGP to build FRR/TI-LFA solution and for TE processes relying on IGP LSDB to build SR policies.

## 8.2. BGP-LS

BGP-LS is expected to be the key service discovery protocol. Every node is expected to advertise via BGP-LS its SRv6 capabilities (e.g. how many SIDs it can insert as part of an T.Encaps behavior) and any locally instantiated SID [I-D.dawra-idr-bgpls-srv6-ext].

## 8.3. BGP IP/VPN/EVPN

The End.DX4, End.DX6, End.DT4, End.DT6, End.DT46, End.DX2, End.DX2V, End.DT2U and End.DT2M SIDs are expected to be signaled in BGP [I-D.dawra-idr-srv6-vpn].

## 8.4. Summary

The following table summarizes which SIDs are signaled in which signaling protocol.





| | IGP | BGP-LS | BGP IP/VPN/EVPN |
|---|---|---|---|
| End (PSP, USP, USD) | X | X | |
| End.X (PSP, USP, USD) | X | X | |
| End.T (PSP, USP, USD) | X | X | |
| End.DX6 | X | X | X |
| End.DX4 | X | X | X |
| End.DT6 | X | X | X |
| End.DT4 | X | X | X |
| End.DT46 | X | X | X |
| End.DX2 | | X | X |
| End.DX2V | | X | X |
| End.DT2U | | X | X |
| End.DT2M | | X | X |

| End.B6.Insert | | X | |
|---|---|---|---|
| End.B6.Insert.Red | | X | |
| End.B6.Encaps | | X | |
| End.B6.Encaps.Red | | X | |
| End.B6.BM | | X | |

Table 1: SRv6 locally instanted SIDs signaling

The following table summarizes which transit capabilities are signaled in which signaling protocol.

| | IGP | BGP-LS | BGP IP/VPN/EVPN |
|---|---|---|---|
| T | | X | |
| T.Insert | X | X | |
| T.Insert.Red | X | X | |
| T.Encaps | X | X | |
| T.Encaps.Red | X | X | |
| T.Encaps.L2 | | X | |
| T.Encaps.L2.Red | | X | |

Table 2: SRv6 transit behaviors signaling

The previous table describes generic capabilities. It does not describe specific instantiated SR policies.

For example, a BGP-LS advertisement of the T capability of node N would indicate that node N supports the basic transit behavior. The T.Insert behavior would describe the capability of node N to perform a T.Insert behavior, specifically it would describe how many SIDs





could be inserted by N without significant performance degradation. Same for T.Encaps (the number is potentially lower as the overhead of the additional outer IP header is accounted).

The reader should also remember that any specific instantiated SR policy is always assigned a Binding SID. They should remember that BSIDs are advertised in BGP-LS as shown in Table 1. Hence, it is

normal that Table 2 only focuses on the generic capabilities related to T.Insert and T.Encaps as Table 1 advertises the specific instantiated BSID properties.

## 9. IANA Considerations

This document requests the following new IANA registries:

- A new top-level registry "Segment-routing with IPv6 dataplane (SRv6) Parameters" to be created under IANA Protocol registries. This registry is being defined to serve as a top-level registry for keeping all other SRv6 sub-registries.

- A sub-registry "SRv6 Endpoint Behaviors" to be defined under top-level "Segment-routing with IPv6 dataplane (SRv6) Parameters" registry. This sub-registry maintains 16-bit identifiers for the SRv6 Endpoint behaviors. The range of the registry is 0-65535 (0x0000 - 0xFFFF) and has the following registration rules and allocation policies:

| Range | Hex | Registration procedure | Notes |
|---|---|---|---|
| 0 | 0x0000 | Reserved | Invalid |
| 1-32767 | 0x0001-0x7FFF | Specification Required | |
| 32768-49151 | 0x8000-0xBFFF | Reserved for experimental use | |
| 49152-65534 | 0xC000-0xFFFE | Reserved for private use | |
| 65535 | 0xFFFF | Reserved | Opaque |

Table 3: SRv6 Endpoint Behaviors Registry

The initial registrations for the "Specification Required" portion of the sub-registry are as follows:





| | | | |
|---|---|---|---|

| Value | Hex | Endpoint behavior | Reference |
|---|---|---|---|
| 1 | 0x0001 | End (no PSP, no USP) | [This.ID] |
| 2 | 0x0002 | End with PSP | [This.ID] |
| 3 | 0x0003 | End with USP | [This.ID] |
| 4 | 0x0004 | End with PSP&USP | [This.ID] |
| 5 | 0x0005 | End.X (no PSP, no USP) | [This.ID] |
| 6 | 0x0006 | End.X with PSP | [This.ID] |
| 7 | 0x0007 | End.X with USP | [This.ID] |
| 8 | 0x0008 | End.X with PSP&USP | [This.ID] |
| 9 | 0x0009 | End.T (no PSP, no USP) | [This.ID] |
| 10 | 0x000A | End.T with PSP | [This.ID] |
| 11 | 0x000B | End.T with USP | [This.ID] |
| 12 | 0x000C | End.T with PSP&USP | [This.ID] |
| 13 | 0x000D | End.B6.Insert | [This.ID] |
| 14 | 0x000E | End.B6.Encaps | [This.ID] |
| 15 | 0x000F | End.BM | [This.ID] |
| 16 | 0x0010 | End.DX6 | [This.ID] |
| 17 | 0x0011 | End.DX4 | [This.ID] |
| 18 | 0x0012 | End.DT6 | [This.ID] |
| 19 | 0x0013 | End.DT4 | [This.ID] |
| 20 | 0x0014 | End.DT46 | [This.ID] |
| 21 | 0x0015 | End.DX2 | [This.ID] |
| 22 | 0x0016 | End.DX2V | [This.ID] |
| 23 | 0x0017 | End.DT2U | [This.ID] |
| 24 | 0x0018 | End.DT2M | [This.ID] |
| 25 | 0x0019 | Reserved | [This.ID] |
| 26 | 0x001A | End.B6.Insert.Red | [This.ID] |
| 27 | 0x001B | End.B6.Encaps.Red | [This.ID] |
| 28 | 0x001C | End with USD | [This.ID] |
| 29 | 0x001D | End with PSP&USD | [This.ID] |
| 30 | 0x001E | End with USP&USD | [This.ID] |
| 31 | 0x001F | End with PSP, USP & USD | [This.ID] |
| 32 | 0x0020 | End.X with USD | [This.ID] |
| 33 | 0x0021 | End.X with PSP&USD | [This.ID] |
| 34 | 0x0022 | End.X with USP&USD | [This.ID] |
| 35 | 0x0023 | End.X with PSP, USP & USD | [This.ID] |
| 36 | 0x0024 | End.T with USD | [This.ID] |
| 37 | 0x0025 | End.T with PSP&USD | [This.ID] |
| 38 | 0x0026 | End.T with USP&USD | [This.ID] |
| 39 | 0x0027 | End.T with PSP, USP & USD | [This.ID] |

Table 4: IETF - SRv6 Endpoint Behaviors

The SRv6 Endpoint Behavior numbers are maintained by the working group until the RFC is published. Note to the RFC Editor: Remove this paragraph before publication.

## 10. Work in progress

We are working on a extension of this document to provide Yang modelling for all the functionality described in this document. This work is ongoing in [I-D.raza-spring-srv6-yang].

## 11. Acknowledgements

The authors would like to acknowledge Stefano Previdi, Dave Barach, Mark Townsley, Peter Psenak, Thierry Couture, Kris Michielsen, Paul Wells, Robert Hanzl, Dan Ye, Gaurav Dawra, Faisal Iqbal, Jaganbabu Rajamanickam, David Toscano, Asif Islam, Jianda Liu, Yunpeng Zhang, Jiaoming Li, Narendra A.K, Mike Mc Gourty, Bhupendra Yadav, Sherif Toulan, Satish Damodaran, John Bettink, Kishore Nandyala Veera Venk, Jisu Bhattacharya and Saleem Hafeez.

## 12. Contributors

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

Dirk Steinberg
Lapishills Consulting Limited
Cyprus

Email: dirk@lapishills.com

Robert Raszuk
Bloomberg LP
United States of America

Email: robert@raszuk.net

Bruno Decraene
Orange
France

Email: bruno.decraene@orange.com

Bart Peirens
Proximus





Belgium

Email: bart.peirens@proximus.com

Hani Elmalky
Ericsson
United States of America

Email: hani.elmalky@gmail.com

Prem Jonnalagadda
Barefoot Networks
United States of America

Email: prem@barefootnetworks.com

Milad Sharif
Barefoot Networks
United States of America

Email: msharif@barefootnetworks.com

David Lebrun
Google
Belgium

Email: dlebrun@google.com

Stefano Salsano
Universita di Roma "Tor Vergata"
Italy

Email: stefano.salsano@uniroma2.it

Ahmed AbdelSalam
Gran Sasso Science Institute
Italy

Email: ahmed.abdelsalam@gssi.it

Gaurav Naik
Drexel University
United States of America

Email: gn@drexel.edu

Arthi Ayyangar
Arista





United States of America

Email: arthi@arista.com

Satish Mynam
Innovium Inc.
United States of America

Email: smynam@innovium.com

Wim Henderickx
Nokia
Belgium

Email: wim.henderickx@nokia.com

Shaowen Ma
Juniper
Singapore

Email: mashao@juniper.net

Ahmed Bashandy
Individual
United States of America

Email: abashandy.ietf@gmail.com

Francois Clad
Cisco Systems, Inc.
France

Email: fclad@cisco.com

Kamran Raza
Cisco Systems, Inc.
Canada

Email: skraza@cisco.com

Darren Dukes
Cisco Systems, Inc.
Canada

Email: ddukes@cisco.com

Patrice Brissete
Cisco Systems, Inc.





Canada

Email: pbrisset@cisco.com

Zafar Ali
Cisco Systems, Inc.
United States of America

Email: zali@cisco.com

## 13. References

### 13.1. Normative References

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-22 (work in progress), August 2019.

[I-D.voyer-6man-extension-header-insertion]
daniel.voyer@bell.ca, d., Leddy, J., Filsfils, C., Dukes, D., Previdi, S., and S. Matsushima, "Insertion of IPv6 Segment Routing Headers in a Controlled Domain", draft-voyer-6man-extension-header-insertion-06 (work in

progress), July 2019.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

## 13.2. Informative References

[I-D.ali-spring-srv6-oam]
Ali, Z., Filsfils, C., Kumar, N., Pignataro, C., faiqbal@cisco.com, f., Gandhi, R., Leddy, J., Matsushima, S., Raszuk, R., daniel.voyer@bell.ca, d., Dawra, G., Peirens, B., Chen, M., and G. Naik, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ali-spring-srv6-oam-02 (work in progress), October 2018.





[I-D.bashandy-isis-srv6-extensions]
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Routing over IPv6 Dataplane", draft-bashandy-isis-srv6-extensions-05 (work in progress), March 2019.

[I-D.dawra-idr-bgpls-srv6-ext]
Dawra, G., Filsfils, C., Talaulikar, K., Chen, M., daniel.bernier@bell.ca, d., Uttaro, J., Decraene, B., and H. Elmalky, "BGP Link State Extensions for SRv6", draft-dawra-idr-bgpls-srv6-ext-06 (work in progress), March 2019.

[I-D.dawra-idr-srv6-vpn]
Dawra, G., Filsfils, C., Dukes, D., Brissette, P., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Decraene, B., Matsushima, S., and S. Zhuang, "BGP

Signaling for SRv6 based Services.", draft-dawra-idr-srv6-vpn-05 (work in progress), October 2018.

[I-D.filsfils-spring-srv6-net-pgm-illustration]
Filsfils, C., Camarillo, P., Li, Z., Matsushima, S., Decraene, B., Steinberg, D., Lebrun, D., Raszuk, R., and J. Leddy, "Illustrations for SRv6 Network Programming", draft-filsfils-spring-srv6-net-pgm-illustration-01 (work in progress), August 2019.

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., daniel.voyer@bell.ca, d., bogdanov@google.com, b., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-03 (work in progress), May 2019.

[I-D.raza-spring-srv6-yang]
Raza, K., Rajamanickam, J., Liu, X., Hu, Z., Hussain, I., Shah, H., daniel.voyer@bell.ca, d., Elmalky, H., Matsushima, S., Horiba, K., and A. Abdelsalam, "YANG Data Model for SRv6 Base and Static", draft-raza-spring-srv6-yang-04 (work in progress), July 2019.

[I-D.xuclad-spring-sr-service-programming]
Clad, F., Xu, X., Filsfils, C., daniel.bernier@bell.ca, d., Li, C., Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and S. Salsano, "Service Programming with Segment Routing", draft-xuclad-spring-sr-service-programming-02 (work in progress), April 2019.





[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, DOI 10.17487/RFC2473, December 1998, <https://www.rfc-editor.org/info/rfc2473>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011,

<https://www.rfc-editor.org/info/rfc6437>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

## Authors' Addresses

Clarence Filsfils (editor)
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Pablo Camarillo Garvia (editor)
Cisco Systems, Inc.
Spain

Email: pcamaril@cisco.com

John Leddy
Individual Contributor
United States of America

Email: john@leddy.net





Daniel Voyer
Bell Canada
Canada

Email: daniel.voyer@bell.ca

Satoru Matsushima
SoftBank
1-9-1,Higashi-Shimbashi,Minato-Ku
Tokyo  105-7322
Japan

Email: satoru.matsushima@g.softbank.co.jp

Zhenbin Li
Huawei Technologies
China

Email: lizhenbin@huawei.com